

eWolf Community

Buku Wajib Programmer:

# Syntax

Kumpulan  
Perintah  
Dasar  
Pemrograman

Disertai  
Contoh  
Penggunaannya

**Buku Wajib Programmer:**

# **Syntax**

**(Kumpulan Perintah Dasar Pemrograman)**

Buku Wajib Programmer:

# Syntax

(Kumpulan Perintah Dasar Pemrograman)

## Undang-Undang RI Nomor 19 Tahun 2002 tentang Hak Cipta

Ketentuan Pidana Pasal 72:

1. Barangsiapa dengan sengaja atau tanpa hak melakukan perbuatan sebagaimana dimaksud dalam Pasal 2 ayat (1) atau Pasal 49 ayat (1) dan ayat (2) dipidana dengan pidana penjara masing-masing paling singkat 1 (satu) bulan dan/atau denda paling sedikit Rp 1.000.000,00 (satu juta rupiah), atau pidana penjara paling lama 7 (tujuh) tahun dan/atau denda paling banyak Rp 5.000.000.000; (lima miliar rupiah).
2. Barangsiapa dengan sengaja menyiarkan, memamerkan, mengedarkan, atau menjual kepada umum suatu ciptaan atau barang hasil pelanggaran Hak Cipta atau hak terkait sebagaimana dimaksud pada ayat (1), dipidana dengan pidana penjara paling lama 5 (lima) tahun dan/atau denda paling banyak Rp 500.000.000,00 (lima ratus juta rupiah).

### **Buku Wajib Programmer: Syntax**

oleh: eWolf Community

© all rights reserved

Hak cipta dilindungi undang-undang

Desain Sampul: Destyan

Tata Letak/Penyunting: xHaz

Diterbitkan oleh:

#### **MediaKom**

Jl. Krasak Timur No. 28A

RT.01/RW.02 Kelurahan Bausasran,

Kecamatan Danurejan, Yogyakarta

Telp. (0274) 6855361, 581788, Faks. (0274) 620879

Email: [penerbitmediakom@gmail.com](mailto:penerbitmediakom@gmail.com)

Website: <http://www.mediakom-penerbit.com>

#### **Hasnul Arifin**

*Buku Wajib Programmer: Syntax/eWolf Community*

—cet. 1 Yogyakarta: Penerbit MediaKom, 2010,

136 hlm; 11 x 17,5 cm

**ISBN (10) 979-877-144-3**

**ISBN (13) 978-979-877-144-6**

1. Komputer

I. Judul

600

#### **Distributor tunggal:**

##### **PT. BUKU SERU**

Jl. Kelapa Hijau No. 22 RT 006/03

Kelurahan Jagakarsa, Kecamatan Jagakarsa

Jakarta 12620

Telp. (021) 7888-1850, Faks. (021) 7888-1860

Cetakan Pertama, 2010

## **Kata Pengantar**

Pemrograman merupakan bahasa wajib yang harus dikuasai oleh seorang *programmer* komputer. Pada dasarnya pemrograman komputer terdiri dari 2 kategori, yaitu pemrograman web dan pemrograman *desktop*. Pemrograman web bertujuan untuk membangun sebuah sistem terpusat melalui jaringan intranet maupun internet. Sedangkan pemrograman *desktop* ditujukan untuk membangun sebuah aplikasi komputer berbasis *desktop*.

Sebagai *programmer*, tentu kedua kategori tersebut harus dikuasai. Namun, rasanya tidak mungkin bila Anda harus menghafal seluruh perintah dasar (syntax) bahasa pemrograman yang begitu banyak.

Buku ini merupakan panduan mudah yang berisi perintah-perintah dasar (syntax) pemrograman yang paling sering digunakan, dimulai dari pemrograman web menggunakan HTML, PHP, juga pemrograman *desktop* menggunakan Visual Basic 6, serta bahasa SQL. Dengan buku ini, maka Anda tidak perlu lagi menghafal seluruh syntax bahasa pemrograman tersebut.

Akhir kata, penulis mengucapkan terimakasih yang sebesar-besarnya kepada semua pihak yang telah membantu dalam terwujudnya buku ini. Kritik dan saran dapat ditujukan melalui email: [eWolfCommunity@yahoo.com](mailto:eWolfCommunity@yahoo.com).

Salam Hormat,

**Penulis**

## Daftar Isi

Kata Pengantar.....5

Daftar Isi.....6

1. Syntax HTML (*Hypertext Markup Language*).....7
2. Syntax PHP (*Hypertext Preprocessor*).....29
3. Syntax SQL (*Structure Query Language*).....105
4. Syntax Visual Basic.....121

# 1

## Syntax HTML (*Hypertext Markup Language*)

### 1. Struktur Dasar

Berikut struktur dasar kode HTML:

```
<html>
  <head>
    <!--Ini adalah bagian header-->
    <title>Judul Web</title>
  </head>

  <body>
    <!--Tempat menuliskan kode-->
  </body>
</html>
```

### 2. <!--...-->

Digunakan untuk menuliskan komentar

Contoh:

```
<!--Ini adalah komentar dan tidak akan
ditampilkan di browser-->
```

### 3. <!DOCTYPE>

Deklarasi ini bukan termasuk tag HTML, tetapi merupakan instruksi untuk *web browser* tentang versi bahasa markup yang digunakan.

Contoh:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml-
ml1/DTD/xhtml1-transitional.dtd">
```

**4. <a>**

Mendefinisikan tag *anchor*. Digunakan untuk membuat *link* ke dokumen lain menggunakan atribut **href** dan juga untuk membuat *bookmark* dokumen menggunakan atribut **name**.

Contoh:

```
<body>
  <a href="http://www.google.com">Kunjungi
    website Google</a>
</body>
```

**5. <abbr>**

Digunakan untuk menampilkan informasi pada saat *pointer mouse* diletakkan pada teks yang ditandai.

Contoh:

```
<body>
  <abbr title="HyperText Markup Language">
    HTML</abbr> adalah kode dasar untuk
    membuat web.
</body>
```

**6. <acronym>**

Digunakan untuk mendefinisikan sebuah singkatan. Informasi akan muncul pada saat *pointer mouse* diletakkan pada teks yang ditandai.

Contoh:

```
<body>
  Hari-hari bersama <acronym title="Facebook">
    FB</acronym> sungguh mengasyikkan
</body>
```

**7. <address>**

Digunakan untuk mendefinisikan informasi kontak pada halaman web. Biasanya digunakan sebagai *signature* pada halaman web.

Contoh:

```
<body>
  <address>
    Hasnul Arifin<br>
    <a href="mailto:hasnul.arifin@gmail.com">
      Kirim Email</a><br>
    Alamat: Yogyakarta<br>
    Telp: +62 xxxxxx
  </address>
</body>
```

**8. <area>**

Digunakan untuk mendefinisikan daerah (area) pada pemetaan gambar.

Contoh:

```

<map name="planetmap">
  <area shape="rect" coords="0,0,82,126"
    href="matahari.htm" alt="Matahari"/>
  <area shape="circle" coords="90,58,3"
    href="mercurius.htm" alt="Mercurius"/>
  <area shape="circle" coords="124,58,8"
    href="venus.htm" alt="Venus"/>
</map>
```

**9. <b>**

Digunakan untuk mencetak tebal suatu teks.

Contoh:

```
<b>Teks tercetak tebal (bold)</b>
```

**10. <base>**

Digunakan untuk mendefinisikan *link default* dalam sebuah halaman web. Tag ini diletakkan pada bagian **head**.

Contoh:

```
<head>
<base href="http://www.w3schools.com/images/" />
<base target="_blank" />
</head>
<body>

<a href="http://www.w3schools.com">
  W3Schools</a>
</body>
```

### 11. <bdo>

Digunakan untuk menentukan arah teks.

Contoh:

```
<bdo dir="ltr">Contoh teks LTR (Left to Right)
</bdo>
<bdo dir="rtl">Contoh teks RTL (Right to Left)
</bdo>
```

### 12. <big>

Digunakan untuk membuat teks berukuran besar.

Contoh:

```
<big>Big text</big>
```

### 13. <blockquote>

Digunakan untuk mendefinisikan pendapat (*quote*) yang panjang. *Browser* Anda akan menambahkan spasi kosong sebelum dan sesudah elemen ini.

Contoh:

```
<blockquote>
  HTML adalah kode dasar untuk membuat web
  HTML adalah kode dasar untuk membuat web
  HTML adalah kode dasar untuk membuat web
</blockquote>
```

### 14. <body>

Digunakan untuk mendefinisikan badan HTML, yang di dalamnya merupakan tempat untuk meletakkan elemen-elemen HTML lainnya, seperti gambar, teks, tabel, *form*, dan lain-lain.

Contoh:

```
<html>
<head>
  <title>Judul Web</title>
</head>
<body>
  <!--Ini adalah bagian badan tempat Anda
  menuliskan kode-->
</body>
</html>
```

### 15. <br>

Digunakan untuk menyisipkan sebuah baris (ganti baris).

Contoh:

```
<body>
  Salah satu aplikasi internet adalah web<br>
  HTML adalah kode dasar untuk membuat web<br>
</body>
```

### 16. <button>

Digunakan untuk mendefinisikan sebuah tombol.

Contoh:

```
<button type="submit">Klik di sini untuk
menyimpan</button>
```

### 17. <caption>

Digunakan untuk mendefinisikan judul pada tabel.

Contoh:

```
<table border="1">
  <caption>Daftar Situs</caption>
```

```

<tr>
  <th>Berita</th>
  <th>Belanja</th>
</tr>
<tr>
  <td>Detik</td>
  <td>Glodokshop</td>
</tr>
</table>

```

## 18. <center>

Digunakan untuk menampilkan teks di tengah halaman.

Contoh:

```

<center>Teks ini akan tampil di tengah
halaman</center>

```

## 19. <cite>

Digunakan untuk mendefinisikan kutipan.

Contoh:

```

<cite>Life is Beautiful</cite> Hidup itu be-
gitu indahnnya

```

## 20. <code>

Digunakan untuk mendefinisikan teks yang berisi kode-kode komputer.

Contoh:

```

<code>Ini adalah teks yang dianggap kode-
kode</code>

```

## 21. <col>

Digunakan untuk mendefinisikan nilai atribut kolom pada tabel.

Contoh:

```

<table width="100%" border="1">
  <col align="left" />
  <col align="left" />

```

```

<col align="right" />
<tr>
  <th>Judul Buku</th>
  <th>ISBN</th>
  <th>Harga</th>
</tr>
<tr>
  <td>Pemrograman Web</td>
  <td>999-888-877-4/5</td>
  <td>45000</td>
</tr>
</table>

```

**Catatan:** pada browser Firefox, hanya atribut **width** pada tag **<col>** yang dapat berfungsi. Atribut lain, seperti **align** pada contoh di atas tidak berfungsi.

## 22. <colgroup>

Digunakan untuk mengelompokkan kolom pada tabel untuk keperluan format tabel.

Contoh:

```

<table width="100%" border="1">
  <colgroup span="2" align="left"></colgroup>
  <colgroup align="right" style="color:#0000FF;">
</colgroup>
  <tr>
    <th>Judul Buku</th>
    <th>ISBN</th>
    <th>Harga</th>
  </tr>
  <tr>
    <td>Pemrograman Web</td>
    <td>999-888-877-4/5</td>
    <td>45000</td>
  </tr>
</table>

```

**Catatan:** browser Firefox, Chrome, dan Safari, hanya mendukung atribut **span** dan **width** pada tag **<colgroup>**. Atribut lain, seperti **align** pada contoh di atas tidak berfungsi.

### 23. <dd>

Digunakan untuk membuat daftar item pada sebuah *definition list* (**<dl>**).

Contoh:

```
<dl>
  <dt>Daftar Menu Makanan</dt>
    <dd>- Ayam goreng</dd>
    <dd>- Ayam bakar</dd>
  <dt>Daftar Menu Minuman</dt>
    <dd>- Macam-macam jus</dd>
    <dd>- Cappucino</dd>
</dl>
```

### 24. <del>

Digunakan untuk mendefinisikan sebuah teks yang telah dihapus (teks tercoret).

Contoh:

```
<b>Khusus promo.. </b><br>
Harga normal <del>Rp. 150.000</del> Harga
diskon <ins>Rp. 130.000</ins>
```

### 25. <dfn>

Digunakan untuk mendefinisikan sebuah istilah.

Contoh:

```
<dfn>Netbook</dfn>, merupakan versi sederhana dari sebuah Notebook.
```

### 26. <dir>

Digunakan untuk memuat daftar judul dari sebuah direktori.

Contoh:

```
<dir>
  <li>html</li>
  <li>xhtml</li>
  <li>css</li>
</dir>
```

### 27. <div>

Digunakan untuk mendefinisikan bagian dari sebuah halaman web. Seringkali digunakan untuk mengelompokkan elemen-elemen HTML dengan format *style* tertentu.

Contoh:

```
<div style="color:#FF0000">
  <h3>Ini Merupakan Judul</h3>
  <p>Ini merupakan paragraf</p>
</div>
```

### 28. <dl>

Digunakan untuk membuat sebuah daftar definisi (*definition list*) yang terdiri dari beberapa item.

Contoh:

```
<dl>
  <dt>Daftar Menu Makanan</dt>
    <dd>- Ayam goreng</dd>
    <dd>- Ayam bakar</dd>
  <dt>Daftar Menu Minuman</dt>
    <dd>- Macam-macam jus</dd>
    <dd>- Cappucino</dd>
</dl>
```

### 29. <dt>

Digunakan untuk membuat daftar item dari sebuah *definition list*.

Contoh:

```
<dl>
  <dt>Daftar Menu Makanan</dt>
```

```

<dd>- Ayam goreng</dd>
<dd>- Ayam bakar</dd>
<dt>Daftar Menu Minuman</dt>
<dd>- Macam-macam jus</dd>
<dd>- Cappucino</dd>
</dl>

```

### 30. <em>

Digunakan untuk menekankan sebuah teks.

Contoh:

<em>HTML</em> adalah kode dasar untuk membuat web

### 31. <fieldset>

Digunakan untuk mengelompokkan beberapa elemen dalam sebuah *form*.

Contoh:

```

<form>
  <fieldset>
    <legend>Buku Tamu</legend>
    Nama: <input type="text" size="30" /><br/>
    Email: <input type="text" size="30" /><br/>
  </fieldset>
</form>

```

### 32. <font>

Digunakan untuk mendefinisikan huruf (jenis, warna, dan ukuran).

Contoh:

```

<font size="3" color="red">Life is Beautiful
</font>
<font face="verdana" color="green">Life is
Beautiful</font>

```

### 33. <form>

Digunakan untuk mendefinisikan sebuah *form* masukan dari pengguna.

Contoh:

```

<form action="action.php" method="get">
  Nama: <input type="text" name="f_nama"/>
  <br>
  Email: <input type="text" name="f_email"/>
  <br>
  <input type="submit" value="Simpan"/>
</form>

```

### 34. <frame>

Digunakan untuk mendefinisikan sebuah jendela (*window*) tertentu dalam sebuah *frameset*.

Contoh:

```

<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm"/>
  <frame src="frame_b.htm"/>
  <frame src="frame_c.htm"/>
</frameset>

```

### 35. <frameset>

Digunakan untuk mendefinisikan sebuah *frameset*. Tiap *frameset* dapat terdiri dari 2 atau lebih *frame*, dan tiap-tiap *frame* memiliki *file* yang berbeda-beda.

Contoh:

```

<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm"/>
  <frame src="frame_b.htm"/>
  <frame src="frame_c.htm"/>
</frameset>

```

### 36. <h1> sampai dengan <h6>

Digunakan untuk mendefinisikan bagian atas (*heading*) teks. Dimulai dari ukuran besar <h1> hingga ukuran kecil <h6>.

Contoh:

```
<h1>HTML adalah kode dasar membuat web</h1>
<h2>HTML adalah kode dasar membuat web</h2>
<h3>HTML adalah kode dasar membuat web</h3>
<h4>HTML adalah kode dasar membuat web</h4>
<h5>HTML adalah kode dasar membuat web</h5>
<h6>HTML adalah kode dasar membuat web</h6>
```

### 37. <head>

Digunakan untuk mendefinisikan bagian atas (*head*) sebuah dokumen HTML. Bagian ini biasanya digunakan untuk mendefinisikan tag <base>, <link>, <meta>, <script>, <style>, dan <title>.

Contoh:

```
<head>
  <title>Judul Web</title>
</head>
```

### 38. <hr>

Digunakan untuk mendefinisikan sebuah garis horisontal dalam halaman HTML.

Contoh:

Salah satu aplikasi internet adalah web<hr>  
HTML adalah kode dasar untuk membuat web

### 39. <i>

Digunakan untuk mencetak miring suatu teks.

Contoh:

```
<i>Teks tercetak miring (italic)</i>
```

### 40. <link>

Digunakan untuk mendefinisikan tautan (*link*) antara halaman dengan sumber lain.

Contoh:

```
<head>
  <link rel="stylesheet" type="text/css"
    href="theme.css" />
</head>
```

### 41. <map>

Digunakan untuk pemetaan gambar.

Contoh:

```

<map name="planetmap">
  <area shape="rect" coords="0,0,82,126"
    href="matahari.htm" alt="Matahari"/>
  <area shape="circle" coords="90,58,3"
    href="mercurius.htm" alt="Mercurius"/>
  <area shape="circle" coords="124,58,8"
    href="venus.htm" alt="Venus"/>
</map>
```

### 42. <menu>

Digunakan untuk membuat daftar menu.

Contoh:

```
<menu>
  <li>html</li>
  <li>xhtml</li>
  <li>css</li>
</menu>
```

### 43. <meta>

Digunakan untuk mendeskripsikan metadata sebuah dokumen HTML.

Contoh:

```
<html>
  <head>
    <meta name="deskripsi" content="Belajar Web"/>
```

```
<meta name="kunci" content="HTML,CSS,XML,
JavaScript"/>
<meta name="pembuat" content="eWolf"/>
<meta http-equiv="Content-Type"
content="text/html; charset=ISO-8859-1"/>
</head>
```

#### 44. <noframes>

Digunakan untuk mengantisipasi jika *browser* tidak mendukung penggunaan *frame*.

Contoh:

```
<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm"/>
  <frame src="frame_b.htm"/>
  <frame src="frame_c.htm"/>
  <noframes>
    Maaf, browser Anda tidak mendukung frame
  </noframes>
</frameset>
```

#### 45. <noscript>

Digunakan untuk mengantisipasi jika *browser* tidak mendukung *client side scripting* atau jika fitur *script* dinonaktifkan.

Contoh:

```
<script type="text/javascript">
  document.write("Ini adalah javascript")
</script>
<noscript>
  Browser Anda tidak mendukung JavaScript,
  periksa kembali pengaturan browser
</noscript>
```

#### 46. <ol>

Digunakan untuk membuat daftar order (*ordered list*). Daftar bisa berupa nomor atau alfabet.

Contoh:

```
<ol>
  <li>web pribadi</li>
  <li>blog</li>
  <li>facebook</li>
</ol>
```

#### 47. <optgroup>

Digunakan untuk mengelompokkan beberapa daftar pilihan.

Contoh:

```
<select>
  <optgroup label="Situs Berita">
    <option value="detik">Detik</option>
    <option value="viva">VIVA News</option>
  </optgroup>
  <optgroup label="Belanja">
    <option value="glodokshop">Glodok Shop
    </option>
    <option value="bhineka">Bhineka</option>
  </optgroup>
</select>
```

#### 48. <option>

Digunakan untuk menampilkan daftar pilihan.

Contoh:

```
<select>
  <option value="detik">Detik</option>
  <option value="viva">VIVA News</option>
  <option value="glodokshop">Glodok Shop
  </option>
  <option value="bhineka">Bhineka</option>
</select>
```

#### 49. <p>

Digunakan untuk membuat paragraf baru.

Contoh:

```
<body>
  Salah satu aplikasi internet adalah web <p>
  HTML adalah kode dasar untuk membuat web
</body>
```

### 50. <param>

Digunakan untuk mendefinisikan parameter atau variabel untuk sebuah elemen objek atau *applet*.

Contoh:

```
<object classid="clsid:F08DF954-8592-11D1-
B16A-00C0F0283628" id="Slider1" width="100"
height="50">
  <param name="BorderStyle" value="1"/>
  <param name="MousePointer" value="0"/>
  <param name="Enabled" value="1"/>
  <param name="Min" value="0"/>
  <param name="Max" value="10"/>
</object>
```

### 51. <pre>

Digunakan untuk mendefinisikan preformat teks.

Contoh:

```
<pre>
Teks dalam tag ini akan tampil seperti
adanya
  dengan spasi dan jarak yang tetap
  tanpa adanya perubahan format
</pre>
```

### 52. <q>

Digunakan untuk mendefinisikan kutipan pada sebuah teks. Tag ini akan menambahkan tanda petik dua (*quote*) pada teks yang ditandai. Hampir semua *browser* mendukung tag ini, kecuali Internet Explorer.

Contoh:

```
Bill gates: <q>If you can't make it good, at
least make it look good.</q>
```

### 53. <s>

Digunakan untuk membuat tanda coretan pada teks seperti halnya pada tag **<del>**.

Contoh:

```
<b>Khusus promo..</b><br>
Harga normal <s>Rp. 150.000</s> Harga diskon
<ins>Rp. 130.000</ins>
```

### 54. <script>

Digunakan untuk mendefinisikan *client side scripting* dalam dokumen HTML.

Contoh:

```
<script type="text/javascript">
  document.write("Ini adalah javascript")
</script>
<noscript>
  Browser Anda tidak mendukung JavaScript,
  periksa kembali pengaturan browser
</noscript>
```

### 55. <select>

Digunakan untuk mendefinisikan daftar pilihan.

Contoh:

```
<select>
  <option value="detik">Detik</option>
  <option value="viva">VIVA News</option>
  <option value="glodokshop">Glodok Shop
  </option>
  <option value="bhineka">Bhineka</option>
</select>
```

**56. <small>**

Digunakan untuk mendefinisikan teks berukuran kecil.

Contoh:

```
<small>HTML adalah kode dasar untuk membuat web</small>
```

**57. <span>**

Digunakan untuk mendefinisikan bagian tertentu dari sebuah dokumen HTML. Tag ini sering digunakan bersama dengan CSS.

Contoh:

```
<head>
<style type="text/css">
    span.blue {color:lightskyblue;
               font-weight:bold}
</style>
</head>
<body>
    Kode dasar untuk membuat web adalah
    <span class="blue">HTML</span>
</body>
```

**58. <strike>**

Digunakan untuk membuat tanda coretan pada teks seperti halnya pada tag <del> atau <s>.

Contoh:

```
<b>Khusus promo..</b><br>
Harga normal <strike>Rp. 150.000</strike>
Harga diskon <ins>Rp. 130.000</ins>
```

**59. <strong>**

Digunakan untuk mempertegas tampilan suatu teks.

Contoh:

```
Kode dasar untuk membuat web adalah
<strong>HTML</strong>
```

**60. <style>**

Digunakan untuk mendefinisikan informasi *style* pada dokumen HTML.

Contoh:

```
<head>
<style type="text/css">
    span.blue {color:lightskyblue;
               font-weight:bold}
</style>
</head>
<body>
    Kode dasar untuk membuat web adalah
    <span class="blue">HTML</span>
</body>
```

**61. <sub>**

Digunakan untuk memformat teks *subscript* (di bawah garis).

Contoh:

Kode kimia air adalah H<sub>2</sub></sub>O

**62. <sup>**

Digunakan untuk memformat teks *superscript* (di atas garis).

Contoh:

Contoh persamaan kuadrat=  
X<sup>2</sup></sup>+Y<sup>2</sup></sup>+XY=0

**63. <table>**

Digunakan untuk membuat tabel dalam halaman web.

Contoh:

```
<table border="1">
    <caption>Daftar Situs</caption>
    <tr>
        <th>Berita</th>
```

```

        <th>Belanja</th>
    </tr>
    <tr>
        <td>Detik</td>
        <td>Glodokshop</td>
    </tr>
</table>

```

**64. <td>**

Digunakan untuk mendefinisikan kolom dalam tabel.

Contoh:

```

<table border="1">
    <caption>Daftar Situs</caption>
    <tr>
        <th>Berita</th>
        <th>Belanja</th>
    </tr>
    <tr>
        <td>Detik</td>
        <td>Glodokshop</td>
    </tr>
</table>

```

**65. <textarea>**

Digunakan untuk mendefinisikan *multiline input*.

Contoh:

```

<textarea rows="2" cols="30">
    Salah satu aplikasi internet adalah web
    HTML adalah kode dasar untuk membuat web
</textarea>

```

**66. <th>**

Digunakan untuk mendefinisikan bagian atas dalam tabel.

Contoh:

```

<table border="1">
    <caption>Daftar Situs</caption>
    <tr>
        <th>Berita</th>
        <th>Belanja</th>
    </tr>
    <tr>
        <td>Detik</td>
        <td>Glodokshop</td>
    </tr>
</table>

```

**67. <title>**

Digunakan untuk mendefinisikan judul pada dokumen HTML.

Contoh:

```

<head>
    <title>Judul Web</title>
</head>

```

**68. <tr>**

Digunakan untuk mendefinisikan baris dalam tabel.

Contoh:

```

<table border="1">
    <caption>Daftar Situs</caption>
    <tr>
        <th>Berita</th>
        <th>Belanja</th>
    </tr>
    <tr>
        <td>Detik</td>
        <td>Glodokshop</td>
    </tr>
</table>

```

### 69. <u>

Digunakan untuk menambahkan garis bawah suatu teks.

Contoh:

```
<u>Teks bergaris bawah(bold)</u>
```

### 70. <ul>

Digunakan untuk membuat daftar yang tak berurut (*unordered list*) menggunakan simbol *bullets*.

Contoh:

```
<ul>
  <li>web pribadi</li>
  <li>blog</li>
  <li>facebook</li>
</ul>
```

# 2

## Syntax PHP (Hypertext Preprocessor)

### 1. Struktur Dasar

Berikut struktur dasar kode HTML:

```
<?php
. . .
?>
```

Dapat juga ditulis:

```
<?
. . .
?>
```

Kode PHP dapat digabung dengan HTML seperti pada contoh berikut:

```
<html>
  <body>
    <?php
      echo "Hello World";
    ?>
  </body>
</html>
```

### 2. /\* ... \*/

Mendefinisikan komentar. Teks yang berada dalam kode ini tidak akan diakses oleh *webserver*.

Contoh:

```
<?php
  /*Ini adalah kolom komentar */
  echo "Hello World";
?>
```

### 3. Variabel

Penulisan variabel dalam PHP diawali dengan symbol \$.

#### Syntax dasar:

```
$nama_variabel = value;
```

Contoh:

```
<?php
$teks="Hello World!";
$bil=10;
?>
```

### 4. if...else

Merupakan statemen kondisi yang digunakan untuk menentukan aksi yang akan dilakukan pada kondisi tertentu.

PHP memiliki 4 macam statemen kondisi, yaitu:

- **if** – digunakan untuk menjalankan kode-kode berikutnya hanya jika kondisi bernilai benar (*true*).

#### Syntax dasar:

```
if (condition) kode yang akan dijalankan
ketika kondisi bernilai benar;
```

Contoh:

```
<?php
$d=date("D");
if ($d=="Fri")
    echo "Selamat weekend";
?>
```

- **if...else** – digunakan untuk menjalankan kode-kode berikutnya jika kondisi bernilai benar (*true*) dan akan menjalankan kode-kode yang lain jika kondisi bernilai salah (*false*).

#### Syntax dasar:

```
if (condition)kode yang akan dijalankan
ketika kondisi bernilai benar;
else
    kode yang akan dijalankan ketika
    kondisi bernilai salah;
```

Contoh:

```
<?php
$d=date("D");
if ($d=="Fri")
    echo "Selamat weekend";
else
    echo "Hari yang menyenangkan";
?>
```

- **if...elseif....else** – digunakan untuk memilih salah satu dari beberapa blok kode yang akan dijalankan.

#### Syntax dasar:

```
if (condition)kode yang akan dijalankan
ketika kondisi bernilai benar;
elseif (condition)kode yang akan dijalan
kan ketika kondisi berikutnya bernilai
benar;
else
    kode yang akan dijalankan ketika
    kondisi bernilai salah;
```

Contoh:

```
<?php
$d=date("D");
if ($d=="Fri")
    echo "Selamat weekend";
elseif ($d=="Sun")
    echo "Senin yang penuh semangat";
else
    echo "Hari yang menyenangkan";
?>
```

- **switch** – digunakan untuk memilih salah satu dari banyak blok kode yang akan dijalankan.

## 5. switch

Merupakan statemen kondisi yang digunakan untuk memilih salah satu dari banyak blok kode yang akan dijalankan.

### Syntax dasar:

```
switch (n)
{
    case label1:
        kode yang akan dijalankan jika n=label1;
        break;
    case label2:
        kode yang akan dijalankan jika n=label2;
        break;
    default:
        kode yang akan dijalankan jika n tidak sama
        dengan label1 dan label2;
}
```

### Contoh:

```
<?php
switch ($x)
{
    case 1:
        echo "Bilangan = 1";
        break;
    case 2:
        echo "Bilangan = 2";
        break;
    case 3:
        echo "Bilangan = 3";
        break;
    default:
        echo "Bilangan selain 1, 2, dan 3";
}
?>
```

## 6. while loop

Merupakan statemen perulangan yang akan menjalankan suatu blok kode selama kondisi terpenuhi (*true*).

### Syntax dasar:

```
while (condition)
{
    kode yang akan dijalankan;
}
```

### Contoh:

```
<?php
$i=1;
while ($i<=5)
{
    echo "Nomor ke " . $i . "<br />";
    $i++;
}
?>
```

### Hasil:

```
Nomor ke 1
Nomor ke 2
Nomor ke 3
Nomor ke 4
Nomor ke 5
```

## 7. do while

Merupakan statemen perulangan yang akan selalu menjalankan sebuah blok kode dan akan berhenti sampai dengan kondisi tidak terpenuhi.

### Syntax dasar:

```
do
{
    kode yang akan dijalankan;
}
while (condition);
```

Contoh:

```
<?php
$i=1;
do
{
    $i++;
    echo "Nomor ke " . $i . "<br />";
}
while ($i<=5);
?>
```

Hasil:

```
Nomor ke 2
Nomor ke 3
Nomor ke 4
Nomor ke 5
Nomor ke 6
```

## 8. for loop

Merupakan statemen perulangan yang digunakan jika Anda sudah mengetahui sebelumnya berapa kali blok kode harus dijalankan.

**Syntax dasar:**

```
for (init; condition; increment)
{
    kode yang akan dijalankan;
}
```

Contoh:

```
<?php
for ($i=1; $i<=5; $i++)
{
    echo "Nomor ke " . $i . "<br />";
}
?>
```

Hasil:

```
Nomor ke 2
Nomor ke 3
Nomor ke 4
Nomor ke 5
Nomor ke 6
```

## 9. foreach

Merupakan statemen perulangan yang digunakan pada tipe *array*.

**Syntax dasar:**

```
foreach ($array as $value)
{
    kode yang akan dijalankan;
}
```

Contoh:

```
<?php
$x=array("satu","dua","tiga");
foreach ($x as $value)
{
    echo $value . "<br />";
}
?>
```

Hasil:

```
satu
dua
tiga
```

## 10. \$\_GET

Digunakan untuk mengumpulkan dan mengambil nilai yang dikirim melalui *form* dengan parameter **method="get"**. Informasi yang dikirim menggunakan metode ini akan terlihat melalui *address bar browser* dan jumlahnya terbatas (maksimum 100 karakter).

Contoh:

Pada *form* pengirim:

```
<form action="welcome.php" method="get">
  Name: <input type="text" name="fnama"/>
  Age: <input type="text" name="umur" />
  <input type="submit" />
</form>
```

Pada *file* penerima:

```
Selamat datang <?php echo $_GET["fnama"]; ?>.<br/>
Usia Anda <?php echo $_GET["umur"]; ?> tahun
```

## 11. checkdate()

Digunakan untuk validasi penanggalan. Fungsi ini akan menghasilkan nilai *true* jika tanggal valid dan *false* jika tanggal tidak valid.

**Syntax dasar:**

```
checkdate(month, day, year)
```

Contoh:

```
<?php
  var_dump(checkdate(12, 31, 2000));
  var_dump(checkdate(2, 29, 2003));
  var_dump(checkdate(2, 29, 2004));
?>
```

Hasil:

```
bool(true) bool(false) bool(true)
```

## 12. date\_default\_timezone\_get()

Digunakan untuk mendapatkan informasi daerah waktu (*timezone*) default yang digunakan oleh seluruh fungsi pada sebuah dokumen PHP.

**Syntax dasar:**

```
date_default_timezone_get(void)
```

Contoh:

```
<?php
  echo(date_default_timezone_get());
?>
```

Hasil:

```
Asia/Krasnoyarsk
```

## 13. date\_default\_timezone\_set()

Digunakan untuk menerapkan (set) daerah waktu (*timezone*) default yang akan digunakan oleh seluruh fungsi pada sebuah dokumen PHP. Daftar *timezone* yang mendukung PHP dapat dilihat pada web <http://www.php.net/manual/en/timezones.php>.

**Syntax dasar:**

```
date_default_timezone_set(timezone)
```

Contoh:

```
<?php
  echo(date_default_timezone_set("Asia/
    Krasnoyarsk"));
?>
```

## 14. date()

Digunakan untuk memformat tanggal dan waktu lokal (*local time*). Referensi parameter dapat dilihat pada web <http://php.net/manual/en/function.date.php>.

**Syntax dasar:**

```
date(format, timestamp)
```

Contoh:

```
<?php
  echo("Tanggal saat ini<br>");
  echo(date("l d F Y h:i:s A") . "<br />");
?>
```

Hasil:

Tanggal saat ini  
Sunday 27 June 2010 01:33:57 AM

## 15. getdate()

Digunakan untuk mendapatkan informasi tanggal dan waktu dalam bentuk *array*. Informasi yang dihasilkan memiliki format:

- [seconds] - detik
- [minutes] - menit
- [hours] - jam
- [mday] - hari dalam sebulan
- [wday] - hari dalam seminggu
- [year] - tahun
- [yday] - hari dalam setahun
- [weekday] - nama hari
- [month] - nama bulan

### Syntax dasar:

```
getdate(timestamp)
```

Contoh:

```
<?php
    $my_t=getdate(date("U"));
    print("$my_t[weekday], $my_t[month],
        $my_t[mday], $my_t[year]");
?>
```

Hasil:

Sunday, June 27, 2010

## 16. idate()

Digunakan untuk memformat tanggal dan waktu lokal (*local time*) ke dalam bentuk bilangan bulat (*integer*).

### Syntax dasar:

```
idate(format,timestamp)
```

Contoh:

```
<?php
    echo(idate("Y"));
?>
```

Hasil:

2010

## 17. strtotime()

Digunakan untuk mengubah tipe *string* menjadi format penanggalan/waktu.

### Syntax dasar:

```
strtotime(time,now)
```

Contoh:

```
<?php
    echo date('F d Y', strtotime('5-1-2010'));
?>
```

Hasil:

January 05 2010

## 18. time()

Digunakan untuk mendapatkan informasi waktu saat ini.

### Syntax dasar:

```
time(void)
```

Contoh:

```
<?php
    $t=time();
    echo(date("D F d Y",$t));
?>
```

Hasil:

Sun June 27 2010

## 19. cal\_days\_in\_month()

Digunakan untuk mendapatkan informasi jumlah hari dalam bulan dan tahun tertentu berdasarkan kalender yang digunakan.

### Syntax dasar:

```
cal_days_in_month(calendar, month, year)
```

### Contoh:

```
<?php
$d=cal_days_in_month(CAL_GREGORIAN,06,2010);
echo("Jumlah hari di Bulan Juni 2010 adalah
$d hari.");
?>
```

### Hasil:

Jumlah hari di Bulan Juni 2010 adalah 30 hari.

## 20. cal\_info()

Digunakan untuk menghasilkan *array* yang berisikan informasi kalender yang digunakan.

### Syntax dasar:

```
cal_info(calendar)
```

**Keterangan:** isi parameter **calendar** dengan tipe kalender yang digunakan, yaitu:

- 0 = CAL\_GREGORIAN
- 1 = CAL\_JULIAN
- 2 = CAL\_JEWISH
- 3 = CAL\_FRENCH

### Contoh:

```
<?php
$calinfo=cal_info(0);
print_r($calinfo);
?>
```

### Hasil:

```
Array ( [months] => Array ( [1] => January [2] =>
February [3] => March [4] => April [5] => May [6]
=> June [7] => July [8] => August [9] => September
[10] => October [11] => November [12] => December
) [abbrevmonths] => Array ( [1] => Jan [2] => Feb
[3] => Mar [4] => Apr [5] => May [6] => Jun [7] =>
Jul [8] => Aug [9] => Sep [10] => Oct [11] => Nov
[12] => Dec ) [maxdaysinmonth] => 31 [calname] =>
Gregorian [calsymbol] => CAL_GREGORIAN )
```

## 21. include() dan require()

Keduanya digunakan untuk menyisipkan dokumen PHP lain ke dalam dokumen yang sedang dibuat. Perbedaan dari kedua fungsi tersebut adalah saat menangani sebuah kesalahan (*error*).

- **include()** akan menghasilkan pesan kesalahan, tetapi kode-kode berikutnya akan tetap dijalankan.

### Contoh:

```
<?php
include("latihan2.php");
echo "Hello World!";
?>
```

- **require()** akan menghasilkan peringatan kesalahan (*fatal error*) dan proses akan stop sampai di situ.

### Contoh:

```
<?php
require("latihan2.php");
echo "Hello World!";
?>
```

## 22. fopen()

Digunakan untuk membuka sebuah *file* atau URL dalam dokumen PHP.

### Syntax dasar:

```
fopen(filename, mode, include_path, context)
```

**Keterangan parameter mode:**

r      Read only.  
 r+     Read/Write.  
 w      Write only.  
 w+     Read/Write.  
 a      Append.  
 a+     Read/Append.  
 x      Write only.  
 x+     Read/Write.

**Contoh:**

```
<?php
$file=fopen("catatan.txt","r");
$file = fopen("/home/test/test.gif","wb");
?>
```

**23. fclose()**

Digunakan untuk membuka sebuah *file* atau URL dalam dokumen PHP.

**Sintax dasar:**

fclose(file)

**Contoh:**

```
<?php
$file=fopen("catatan.txt","r");
fclose($file);
?>
```

**24. fgets()**

Digunakan untuk membaca baris perbaris *file* yang dibuka.

**Sintax dasar:**

fgets(file, length)

**Contoh:**

```
<?php
$file = fopen("catatan.txt","r");
echo fgets($file);
fclose($file);
?>
```

**25. feof()**

Digunakan untuk memeriksa akhiran dari suatu *file* (eof).

**Sintax dasar:**

feof(file)

**Contoh:**

```
<?php
$file = fopen("catatan.txt", "r");
while(! feof($file))
{
    echo fgets($file). "<br />";
}
fclose($file);
?>
```

**26. fgetc()**

Digunakan untuk membaca perkarakter *file* yang dibuka.

**Sintax dasar:**

fgetc(file)

**Contoh:**

```
<?php
$file = fopen("catatan.txt","r");
echo fgetc($file);
fclose($file);
?>
```

**27. file\_exists()**

Digunakan untuk memeriksa ada atau tidaknya sebuah *file*.

**Sintax dasar:**

file\_exists(path)

**Contoh:**

```
<?php
$nama_file = 'path/catatan.txt';
if (file_exists($nama_file)) {
    echo "File $nama_file ditemukan";
}
```

```

} else {
    echo "File $nama_file tidakn ditemukan";
}
?>

```

## 28. fputs() dan fwrite()

Digunakan untuk menambahkan teks pada *file* yang dibuka. Keduanya memiliki fungsi yang sama. Kedua fungsi ini akan menghasilkan keluaran berupa jumlah karakter yang telah ditambahkan pada sebuah *file*.

### Syntax fputs():

```
fputs(file, string, length)
```

Contoh:

```

<?php
$file = fopen("catatan.txt", "w");
echo fputs($file, "PHP merupakan server side
scripting");
fclose($file);
?>

```

### Syntax fwrite():

```
fwrite(file, string, length)
```

Contoh:

```

<?php
$file = fopen("catatan.txt", "w");
echo fwrite($file, "PHP merupakan server side
scripting");
fclose($file);
?>

```

## 29. file()

Digunakan untuk membaca *file* dan memasukkannya ke dalam *array*.

### Syntax dasar:

```
file(path, include_path, context)
```

Contoh:

```

<?php
    print_r(file("catatan.txt"));
?>

```

Hasil:

```
Array ([0] => PHP merupakan server side
scripting)
```

## 30. fseek()

Digunakan pada proses pencarian dalam sebuah *file*. Fungsi ini akan memindahkan *pointer* pada posisi yang ditentukan saat proses pencarian berlangsung.

### Syntax dasar:

```
fseek(file, offset, whence)
```

Contoh:

```

<?php
$file = fopen("catatan.txt", "r");
// membaca dari baris pertama
fgets($file);
// menggerakkan pointer ke awal file
fseek($file, 0);
?>

```

## 31. ftell()

Digunakan untuk mengetahui posisi *pointer* saat ini dalam sebuah *file*.

### Syntax dasar:

```
ftell(file)
```

Contoh:

```

<?php
$file = fopen("catatan.txt", "r");
// menampilkan posisi pointer saat ini
echo ftell($file);

```

```
// mengubah posisi pointer
fseek($file,"15");
//menampilkan kembali posisi pointer saat ini
echo "<br />" . ftell($file);
fclose($file);
?>
```

Hasil:

```
0
15
```

### 32. fstat()

Digunakan untuk mengetahui informasi dari sebuah *file*.

**Syntax dasar:**

```
fstat (file)
```

Contoh:

```
<?php
$file = fopen("catatan.txt","r");
print_r(fstat($file));
fclose($file);
?>
```

### 33. fread()

Digunakan untuk membaca informasi tertentu dari sebuah *file*.

**Syntax dasar:**

```
fread (file, length)
```

Contoh:

```
<?php
$file = fopen("catatan.txt","r");
fread($file,"10");
fclose($file);
?>
```

**Keterangan:**

Contoh di atas akan membaca 10 bytes dari *file catatan.txt*.

### 34. copy()

Digunakan untuk menyalin (*copy*) sebuah *file* menjadi *file* lain.

**Syntax dasar:**

```
copy (file, to_file)
```

Contoh:

```
<?php
echo copy("catatan.txt","catatan2.txt");
?>
```

**Keterangan:**

Contoh di atas akan menyalin *file catatan.txt* menjadi *file catatan2.txt*.

### 35. unlink()

Digunakan untuk menghapus sebuah *file*.

**Syntax dasar:**

```
unlink (filename, context)
```

Contoh:

```
<?php
$file = "catatan2.txt";
if (!unlink($file))
{
    echo ("$file tidak dapat dihapus");
}
else
{
    echo ("$file sudah dihapus");
}
?>
```

### 36. chmod()

Digunakan untuk mengubah hak akses sebuah *file*.

**Syntax dasar:**

```
chmod(file, mode)
```

**Contoh:**

```
<?php
// Akses Read dan Write hanya untuk user
pembuatnya
chmod("catatan.txt", 0600);
// Akses Read dan Write hanya untuk user
pembuatnya dan hanya akses Read untuk
user lainnya
chmod("catatan.txt", 0644);
?>
```

### 37. dirname()

Digunakan untuk menghasilkan informasi nama direktori dari sebuah *path*.

**Syntax dasar:**

```
dirname(path)
```

**Contoh:**

```
<?php
echo dirname("/test/catatan.txt");
?>
```

**Hasil:**

```
/test
```

### 38. disk\_free\_space()

Digunakan untuk menampilkan informasi sisa *space* dalam satuan bytes untuk *harddisk* atau direktori tertentu.

**Syntax dasar:**

```
Disk_free_space(directory)
```

**Contoh:**

```
<?php
echo disk_free_space("C:");
?>
```

**Hasil:**

```
20767215616
```

### 39. basename()

Digunakan untuk menampilkan informasi nama *file* dari *path* tertentu.

**Syntax dasar:**

```
basename(path, suffix)
```

**Contoh:**

```
<?php
$path = "/test/catatan.txt";
//Menampilkan nama file beserta ekstension
echo basename($path) . "<br/>";
//Menampilkan nama file tanpa ekstension
echo basename($path, ".txt");
?>
```

**Hasil:**

```
catatan.txt
catatan
```

### 40. is\_dir()

Digunakan untuk memeriksa apakah *file* yang dimaksud adalah sebuah direktori.

**Syntax dasar:**

```
is_dir(file)
```

**Contoh:**

```
<?php
$file = "images";
if(is_dir($file))
```

```

{
    echo ("$file adalah sebuah direktori");
}
else
{
    echo ("$file tersebut bukanlah direktori");
}
?>

```

#### 41. is\_executable()

Digunakan untuk memeriksa apakah *file* yang dimaksud adalah sebuah *file executable*.

**Syntax dasar:**

```
is_executable(file)
```

Contoh:

```

<?php
$file = "setup.exe";
if(is_executable($file))
{
    echo ("$file merupakan file executable");
}
else
{
    echo ("$file bukan file executable");
}
?>

```

#### 42. is\_readable()

Digunakan untuk memeriksa apakah *file* yang dimaksud adalah sebuah *file* yang dapat dibaca.

**Syntax dasar:**

```
is_readable(file)
```

Contoh:

```

<?php
$file = "catatan.txt";

```

```

if(is_readable($file))
{
    echo ("$file dapat dibaca");
}
else
{
    echo ("$file tidak dapat dibaca");
}
?>

```

#### 43. is\_writable()

Digunakan untuk memeriksa apakah *file* yang dimaksud adalah sebuah *file* yang dapat ditulis.

**Syntax dasar:**

```
is_writable(file)
```

Contoh:

```

<?php
$file = "catatan.txt";
if(is_writable($file))
{
    echo ("$file dapat ditulis");
}
else
{
    echo ("$file tidak dapat ditulis");
}
?>

```

#### 44. pathinfo()

Digunakan untuk mendapatkan informasi mengenai *path* tertentu.

**Syntax dasar:**

```
pathinfo(path,options)
```

Contoh:

```
<?php
    print_r(pathinfo("/testweb/test.txt"));
?>
```

Hasil:

```
Array ( [dirname] => /test [basename] =>
catatan.txt [extension] => txt [filename] =>
catatan )
```

#### 45. mkdir()

Digunakan untuk membuat direktori.

**Syntax dasar:**

```
mkdir(path,mode,recursive,context)
```

Contoh:

```
<?php
    mkdir("images");
?>
```

#### 46. rename()

Digunakan untuk mengubah nama *file* atau direktori.

**Syntax dasar:**

```
rename(oldname,newname,context)
```

Contoh:

```
<?php
    rename("images","gambar");
?>
```

#### 47. rmdir()

Digunakan untuk menghapus sebuah direktori kosong.

**Syntax dasar:**

```
rmdir(dir,context)
```

Contoh:

```
<?php
    $path = "images";
    if(!rmdir($path)
    {
        echo ("$path tidak dapat dihapus");
    }
?>
```

#### 48. readfile()

Digunakan untuk membaca *file* dan menuliskannya di *folder output*.

**Syntax dasar:**

```
readfile(filename,include_path,context)
```

Contoh:

```
<?php
    echo readfile("catatan.txt");
?>
```

Hasil:

PHP merupakan server side scripting35

**Keterangan:**

Contoh di atas akan membaca *file catatan.txt* dan menampilkan isi beserta ukuran *file*-nya.

#### 49. setcookie()

Digunakan untuk menerapkan *cookie* pada dokumen PHP.

**Syntax dasar:**

```
setcookie(name, value, expire, path, domain)
```

Contoh:

```
<?php
    setcookie("user", "Arini", time()+3600);
?>
```

**Keterangan:**

Contoh di atas digunakan untuk menerapkan *cookie user*.  
Berikut contoh untuk mengambil/mendapatkan nilai *cookie*:

```
<?php
    if (isset($_COOKIE["user"]))
        echo "Welcome " . $_COOKIE["user"] .
            "<br />";
    else
        echo "Welcome guest!<br />";
?>
```

**50. session\_start()**

Digunakan untuk menyimpan data *session*.

**Syntax dasar:**

```
session_start()
```

**Contoh:**

```
<?php
    session_start();
    // menyimpan data session
    $_SESSION['views']=1;
?>
```

**Keterangan:**

Contoh di atas digunakan untuk membuat *session views*.  
Berikut contoh untuk mengambil/mendapatkan nilai *session* yang telah dibuat:

```
<html>
    <body>
        <?php
            //mengambil data session
            echo "Pageviews=" . $_SESSION['views'];
        ?>
    </body>
</html>
```

**51. session\_destroy()**

Digunakan untuk menghapus *session* yang telah dibuat.

**Syntax dasar:**

```
session_destroy()
```

**Contoh:**

```
<?php
    session_destroy();
?>
```

**52. unset()**

Seperti halnya fungsi **session\_destroy()**, fungsi **unset()** juga digunakan untuk menghapus *session* yang telah dibuat.

**Syntax dasar:**

```
unset(session)
```

**Contoh:**

```
<?php
    unset($_SESSION['views']);
?>
```

**53. mail()**

Digunakan untuk mengirim *email* melalui kode PHP.

**Syntax dasar:**

```
mail(to,subject,message,headers,parameters)
```

**Contoh:**

```
<?php
    $to = "ewolfcommunity@yahoo.com";
    $subject = "Test mail";
    $message = "Hallo.. apakabar..";
    $from = "hasnul.arifin@gmail.com";
    $headers = "From: $from";
    mail($to,$subject,$message,$headers);
    echo "Email terkirim..";
?>
```

**54. die()**

Digunakan untuk menangani kesalahan (*error*).

Contoh:

```
<?php
    if (!file_exists("catatan.txt"))
    {
        die("File not found");
    }
    else
    {
        $file=fopen("catatan.txt","r");
    }
?>
```

**55. try, throw, dan catch**

Digunakan untuk menghindari kesalahan.

Contoh:

```
<?php
//membuat fungsi dengan exception
function checkNum($number)
{
    if($number>1)
    {
        throw new Exception("nilai harus kurang
                               atau sama dengan 1");
    }
    return true;
}

//exception dalam blok "try"
try
{
    checkNum(2);
}
//Jika exception diabaikan, teks tidak terlihat
echo 'Jika Anda melihat teks ini, berarti
      nilai kurang atau sama dengan 1';
}
```

```
// exception catch
catch(Exception $e)
{
    echo 'Message: `.`.$e->getMessage();
}
?>
```

**56. set\_exception\_handler()**

Digunakan untuk menangani *exception* selama program berjalan.

Syntax dasar:

```
set_exception_handler(exception_function)
```

Contoh:

```
<?php
function myException($exception)
{
    echo "<b>Exception:</b> " , $exception->
    getMessage();
}
set_exception_handler('myException');
throw new Exception('Uncaught Exception
occurred');
?>
```

**57. error\_log()**

Digunakan untuk mengirimkan catatan kesalahan (*error*) ke server atau tujuan lainnya.

Syntax dasar:

```
error_log(error,type,destination,headers)
```

Contoh:

```
<?php
$test=2;
if ($test>1) {
    error_log("error log",1,"eWolfCommunity@
yahoo.com","From: hasnul.arifin@gmail.com");
}
?>
```

**58. filter\_var()**

Digunakan untuk memfilter variabel dengan spesifikasi filter tertentu.

**Syntax dasar:**

```
filter_var(variable, filter, options)
```

Contoh:

```
<?php
if(!filter_var("eWolfCommunity@yahoo.com",
    FILTER_VALIDATE_EMAIL)) {
    echo("Alamat e-mail tidak valid");
}
else {
    echo("Alamat e-mail valid");
}
?>
```

**59. filter\_input()**

Digunakan sebagai validasi variabel yang bersumber dari beberapa sumber, seperti *user input*, *cookie*, dan *session*.

**Syntax dasar:**

```
filter_input(input_type, variable, filter, options)
```

Contoh:

```
<?php
if (!filter_input(INPUT_POST, 'email',
    FILTER_VALIDATE_EMAIL))
{
    echo("Alamat e-mail tidak valid");
}
else
{
    echo("Alamat e-mail valid");
}
?>
```

**60. mysql\_connect()**

Digunakan untuk menghubungkan dokumen dengan *database* MySQL.

**Syntax dasar:**

```
mysql_connect(servername,username,password)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
    "12345678");
if (!$con)
{
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
// kode-kode lainnya
?>
```

**61. mysql\_close()**

Digunakan untuk memutus hubungan dengan *database* MySQL.

**Syntax dasar:**

```
mysql_close(connection)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
    "12345678");
if (!$con)
{
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
// kode-kode lainnya
mysql_close($con);
?>
```

## 62. mysql\_affected\_rows()

Digunakan untuk mendapatkan jumlah baris/*record* yang dihasilkan dari operasi MySQL sebelumnya.

### Syntax dasar:

```
mysql_affected_rows(connection)
```

### Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_user",
    "mysql_pwd");
if (!$con)
{
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
mysql_select_db("mydb");
mysql_query("DELETE FROM mytable WHERE id < 5");
$rc = mysql_affected_rows();
echo "jumlah record terhapus: " . $rc;
mysql_close($con);
?>
```

### Hasil:

jumlah record terhapus: 5

## 63. mysql\_fetch\_array()

Digunakan untuk mendapatkan *array* baris/*record* dari suatu *recordset*.

### Syntax dasar:

```
mysql_fetch_array(data,array_type)
```

### Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_
user","mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
```

```
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu WHERE
    nama='has'";
$result = mysql_query($sql,$con);
print_r(mysql_fetch_array($result));
mysql_close($con);
?>
```

## 64. mysql\_fetch\_assoc()

Digunakan untuk mendapatkan *array* baris/*record* dari suatu *recordset*.

### Syntax dasar:

```
mysql_fetch_assoc(data)
```

### Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_
user","mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
$db_selected = mysql_select_db("test_
db",$con);
$sql = "SELECT * from bukutamu WHERE
    nama='has'";
$result = mysql_query($sql,$con);
print_r(mysql_fetch_assoc($result));
mysql_close($con);
?>
```

## 65. mysql\_fetch\_field()

Digunakan untuk mendapatkan sebuah objek yang mengandung informasi *field* dari sebuah *recordset*.

### Syntax dasar:

```
mysql_fetch_field(data,field_offset)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql,$con);
while ($property = mysql_fetch_field($result))
{
    echo "Field name: " . $property->name . "<br/>";
    echo "Table name: " . $property->table . "<br/>";
    echo "Default value: " . $property->def . "<br/>";
    echo "Max length: " . $property->max_length .
    "<br/>";
    echo "Not NULL: " . $property->not_null . "<br/>";
    echo "Primary Key: " . $property->primary_key .
    "<br/>";
    echo "Unique Key: " . $property->unique_key .
    "<br/>";
    echo "Multiple Key: " . $property->multiple_key .
    "<br/>";
    echo "Numeric Field: " . $property->numeric .
    "<br/>";
    echo "BLOB: " . $property->blob . "<br/>";
    echo "Field Type: " . $property->type . "<br/>";
    echo "Unsigned: " . $property->unsigned . "<br/>";
    echo "Zero-filled: " . $property->zerofill . "<br/>";
}
mysql_close($con);
?>
```

## 66. mysql\_fetch\_lengths()

Digunakan untuk mendapatkan panjang baris/record pada setiap isi field.

**Syntax dasar:**

```
mysql_fetch_lengths(data)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu WHERE
        nama='has'";
$result = mysql_query($sql,$con);
print_r(mysql_fetch_row($result));
print_r(mysql_fetch_lengths($result));
mysql_close($con);
?>
```

## 67. mysql\_fetch\_object()

Digunakan untuk menghasilkan baris/record dari recordset sebagai sebuah objek.

**Syntax dasar**

```
mysql_fetch_object(data)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
```

```

$db_selected = mysql_select_db("test_db", $con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql, $con);
while ($row = mysql_fetch_object($result))
{
    echo $row->Nama . "<br />";
}
mysql_close($con);
?>

```

## 68. mysql\_fetch\_row()

Digunakan untuk menghasilkan *array* numerik dari baris/*record* pada sebuah *recordset*.

**Syntax dasar:**

```
mysql_fetch_row(data)
```

Contoh:

```

<?php
$con = mysql_connect("localhost", "mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db", $con);
$sql = "SELECT * from bukutamu WHERE
    nama='has'";
$result = mysql_query($sql, $con);
print_r(mysql_fetch_row($result));
mysql_close($con);
?>

```

## 69. mysql\_field\_len()

Digunakan untuk mendapatkan informasi panjang maksimum *field* dalam sebuah *recordset*.

**Syntax dasar:**

```
mysql_field_len(data, field_offset)
```

Contoh:

```

<?php
$con = mysql_connect("localhost", "mysql_user",
"mysql_pwd");
if (!$con)
{
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db", $con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql, $con);
$length = mysql_field_len($result, 0);
echo $length;
mysql_close($con);
?>

```

## 70. mysql\_field\_name()

Digunakan untuk mendapatkan informasi nama *field* dalam sebuah *recordset*.

**Syntax dasar:**

```
mysql_field_name(data, field_offset)
```

Contoh:

```

<?php
$con = mysql_connect("localhost", "mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db", $con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql, $con);
$name = mysql_field_name($result, 0);
echo $name;
mysql_close($con);
?>

```

**71. mysql\_field\_seek()**

Digunakan untuk melompat ke *field* tertentu dalam sebuah *recordset*.

**Syntax dasar:**

```
mysql_field_seek(data,field_offset)
```

**Contoh:**

```
<?php
$con = mysql_connect("localhost","mysql_user",
"mysql_pwd");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql,$con);
// melompat ke field 3
mysql_field_seek($result,3);
print_r(mysql_fetch_field($result));
mysql_close($con);
?>
```

**72. mysql\_get\_client\_info()**

Digunakan untuk mendapatkan informasi tentang MySQL *client*.

**Syntax dasar:**

```
mysql_get_client_info()
```

**Contoh:**

```
<?php
echo "MySQL client info: " . mysql_get_
client_info();
?>
```

**73. mysql\_get\_host\_info()**

Digunakan untuk mendapatkan informasi tentang MySQL *client*.

**Syntax dasar:**

```
mysql_get_host_info(connection)
```

**Contoh:**

```
<?php
$con = mysql_connect("localhost", "root",
"12345678");
echo "MySQL host info: " . mysql_get_host_
info($con);
?>
```

**74. mysql\_get\_server\_info()**

Digunakan untuk mendapatkan informasi tentang MySQL *server*.

**Syntax dasar:**

```
mysql_get_server_info(connection)
```

**Contoh:**

```
<?php
$con = mysql_connect("localhost", "root",
"12345678");
echo "MySQL host info: " . mysql_get_server_
info($con);
?>
```

**75. mysql\_info()**

Digunakan untuk mendapatkan informasi tentang *query* terakhir.

**Syntax dasar:**

```
mysql_info(connection)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "INSERT INTO bukutamu VALUES
('arini','ramadhani','Jogja','19')";
$result = mysql_query($sql,$con);
$info = mysql_info($con);
echo $info;
mysql_close($con);
?>
```

## 76. mysql\_num\_fields()

Digunakan untuk mendapatkan informasi jumlah *field* pada suatu *recordset*.

**Syntax dasar:**

```
mysql_num_fields(data)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql,$con);
echo mysql_num_fields($result);
mysql_close($con);
?>
```

## 77. mysql\_num\_rows()

Digunakan untuk mendapatkan informasi jumlah baris/*record* pada suatu *recordset*.

**Syntax dasar:**

```
mysql_num_rows(data)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
    mysql_error());
}
$db_selected = mysql_select_db("test_db",$con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql,$con);
echo mysql_num_rows($result);
mysql_close($con);
?>
```

## 78. mysql\_ping()

Digunakan untuk memeriksa koneksi server dan akan mencoba untuk melakukan koneksi ulang jika koneksi terputus.

**Syntax dasar:**

```
mysql_ping(connection)
```

Contoh:

```
if (!mysql_ping($conn)) {
    echo "Koneksi server terputus";
    exit;
}
```

## 79. mysql\_query()

Digunakan untuk menjalankan *query* pada *database* MySQL.

**Syntax dasar:**

```
mysql_query(query,connection)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
$sql = "SELECT * from bukutamu";
mysql_query($sql,$con);
// tempat kode-kode lainnya
mysql_close($con);
?>
```

## 80. mysql\_result()

Digunakan untuk mendapatkan nilai dari *field* dalam sebuah *recordset*.

**Syntax dasar:**

```
mysql_result(data,row,field)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
$db_selected = mysql_select_db("test_db", $con);
$sql = "SELECT * from bukutamu";
$result = mysql_query($sql,$con);
echo mysql_result($result,0);
mysql_close($con);
?>
```

## 81. mysql\_select\_db()

Digunakan untuk menentukan (set) *database* MySQL yang aktif.

**Syntax dasar:**

```
mysql_select_db(database,connection)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
$db_selected = mysql_select_db("test_db", $con);
if (!$db_selected) {
    die ("Database tidak dapat digunakan : " .
        mysql_error());
}
mysql_close($con);
?>
```

## 82. mysql\_stat()

Digunakan untuk menghasilkan informasi status server terkini.

**Syntax dasar:**

```
mysql_stat(connection)
```

Contoh:

```
<?php
$con = mysql_connect("localhost","root",
"12345678");
if (!$con) {
    die('Database tidak dapat terhubung: ' .
        mysql_error());
}
```

```
$db_selected = mysql_select_db("test_db", $conn);
echo mysql_stat();
?>
```

### 83. ftp\_connect()

Digunakan untuk membuka koneksi FTP ke server.

**Syntax dasar:**

```
ftp_connect(host,port,timeout)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
?>
```

### 84. ftp\_close()

Digunakan untuk menutup koneksi FTP.

**Syntax dasar:**

```
ftp_close(ftp_connection)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
//tempat kode-kode lain untuk dijalankan
ftp_close($conn);
?>
```

### 85. ftp\_login()

Digunakan untuk mengambil sebuah *file* dari FTP server dan menyimpannya ke *drive* lokal.

**Syntax dasar:**

```
ftp_login(ftp_connection,username,password)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
//tempat kode-kode lain untuk dijalankan
ftp_close($conn);
?>
```

### 86. ftp\_chdir()

Digunakan untuk mengubah lokasi direktori saat ini ke direktori lain.

**Syntax dasar:**

```
ftp_chdir(ftp_connection,directory)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
//menampilkan direktori saat ini
echo "Dir: ".ftp_pwd($conn);
echo "<br />";
//mengubah letak direktori ke direktori gambar
ftp_chdir($conn,"gambar");
echo "Dir: ".ftp_pwd($conn);
ftp_close($ftp_server);
?>
```

### 87. ftp\_chmod()

Digunakan untuk menerapkan hak akses pada *file* tertentu di FTP server.

**Syntax dasar:**

```
ftp_chmod(ftp_connection,mode,file)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"user","pass");
// Memberi hak akses Read dan Write utk owner
ftp_chmod($conn,"0600","test.txt");
// Memberi hak akses Read dan Write untuk
owner & Read untuk pengguna lain
ftp_chmod($conn,"0644","test.txt");
/* Memberi seluruh hak akses untuk owner,
Read & Execute untuk pengguna lain */
ftp_chmod($conn,"0755","test.txt");
/* Memberi seluruh hak akses untuk owner,
Read untuk grup owner */
ftp_chmod($conn,"0740","test.txt");
ftp_close($conn);
?>
```

## 88. ftp\_delete()

Digunakan untuk menghapus *file* di FTP server.

**Syntax dasar:**

```
ftp_delete(ftp_connection,file)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
echo ftp_delete($conn,"test.txt");
ftp_close($conn);
?>
```

## 89. ftp\_exec()

Digunakan untuk menjalankan program atau perintah di FTP server.

**Syntax dasar:**

```
ftp_exec(ftp_connection,command)
```

Contoh:

```
<?php
$command = "ls-al > test.txt";
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
if (ftp_exec($conn,$command))
{
    echo "Perintah sukses dijalankan";
}
else
{
    echo "Perintah gagal dijalankan";
}
ftp_close($conn);
?>
```

## 90. ftp\_fget()

Digunakan untuk mengambil sebuah *file* dari FTP server dan menyimpannya untuk selanjutnya dijalankan.

**Syntax dasar:**

```
ftp_fget(ftp_connection,local,remote,mode,
resume)
```

Contoh:

```
<?php
$source = "source.txt";
$target = fopen("target.txt", "w");
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
ftp_fget($conn,$target,$source,FTP_ASCII);
ftp_close($conn);
?>
```

**91. ftp\_fput()**

Digunakan untuk *upload file* ke FTP server.

**Syntax dasar:**

```
ftp_fput(ftp_connection, remote, local, mode,
resume)
```

**Contoh:**

```
<?php
$source = "source.txt";
$conn = ftp_connect("ftp.ftpkku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
echo ftp_fput($conn, "target.txt", $source,
FTP_ASCII);
ftp_close($conn);
?>
```

**92. ftp\_get()**

Digunakan untuk mengambil sebuah *file* dari FTP server dan menyimpannya ke *drive* lokal.

**Syntax dasar:**

```
ftp_get(ftp_connection, local, remote, mode,
resume)
```

**Contoh:**

```
<?php
$conn = ftp_connect("ftp.ftpkku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
echo ftp_get($conn, "target.txt", "source.txt",
FTP_ASCII);
ftp_close($conn);
?>
```

**93. ftp\_put()**

Digunakan untuk *upload file* dari *drive* lokal ke FTP server.

**Syntax dasar:**

```
ftp_put(ftp_connection, remote, local, mode,
resume)
```

**Contoh:**

```
<?php
$conn = ftp_connect("ftp.ftpkku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
echo ftp_put($conn, "target.txt", "source.txt",
FTP_ASCII);
ftp_close($conn);
?>
```

**94. ftp\_mkdir()**

Digunakan untuk membuat direktori di FTP server

**Syntax dasar:**

```
ftp_mkdir(ftp_connection, dir)
```

**Contoh:**

```
<?php
$conn = ftp_connect("ftp.ftpkku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
echo ftp_mkdir($conn, "direktoribaru");
ftp_close($conn);
?>
```

**95. ftp\_pasv()**

Digunakan untuk *setting mode passive on/off*

**Syntax dasar:**

```
ftp_pasv(ftp_connection, mode)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
ftp_pasv($conn,TRUE);
ftp_close($conn);
?>
```

Keterangan:

- TRUE = mode *passive on*.
- FALSE = mode *passive off*.

## 96. ftp\_nlist()

Digunakan untuk menampilkan daftar *file* pada direktori tertentu di FTP server.

**Syntax dasar:**

```
ftp_nlist(ftp_connection,dir)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
print_r(ftp_nlist($conn,"gambar"));
ftp_close($conn);
?>
```

## 97. ftp\_pwd()

Digunakan untuk mendapatkan informasi direktori yang aktif saat ini.

**Syntax dasar:**

```
ftp_pwd(ftp_connection)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
// menampilkan direktori saat ini
echo ftp_pwd($conn);
ftp_close($conn);
?>
```

## 98. ftp\_quit()

Digunakan untuk menutup koneksi FTP.

**Syntax dasar:**

```
ftp_quit(ftp_connection)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
// beberapa kode
ftp_quit($conn);
?>
```

## 99. ftp\_rename()

Digunakan untuk mengubah nama *file* di server.

**Syntax dasar:**

```
ftp_rename(ftp_connection,from,to)
```

Contoh:

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn,"admin","12345678");
ftp_rename($conn,"oldname.txt","newname.txt");
ftp_close($conn);
?>
```

**100. ftp\_rmdir()**

Digunakan untuk menghapus direktori di FTP server.

**Syntax dasar:**

```
ftp_rmdir(ftp_connection, dir)
```

**Contoh:**

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
ftp_rmdir($conn, "testdir");
ftp_close($conn);
?>
```

**101. ftp\_ssl\_connect()**

Digunakan untuk membuka koneksi SSL-FTP.

**Syntax dasar:**

```
ftp_ssl_connect(host, port, timeout)
```

**Contoh:**

```
<?php
$conn = ftp_ssl_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
?>
```

**102. ftp\_size()**

Digunakan untuk mendapatkan informasi ukuran *file* di FTP server.

**Syntax dasar:**

```
ftp_size(ftp_connection, file)
```

**Contoh:**

```
<?php
$conn = ftp_connect("ftp.ftpku.com") or
die("Tidak dapat terhubung");
ftp_login($conn, "admin", "12345678");
```

```
echo ftp_size($conn, "test.txt");
ftp_close($conn);
?>
```

**103. abs()**

Digunakan untuk menghasilkan nilai absolut dari sebuah bilangan.

**Syntax dasar:**

```
abs(x)
```

**Contoh:**

```
<?php
echo (abs(5.2) . "<br />");
echo (abs(-2) . "<br />");
echo (abs(2));
?>
```

**Hasil:**

```
5.2
2
2
```

**104. sin()**

Digunakan untuk menghasilkan nilai *sine* dari bilangan.

**Syntax dasar:**

```
sin(x)
```

**Contoh:**

```
<?php
echo (sin(3) . "<br />");
echo (sin(-3) . "<br />");
echo (sin(0) . "<br />");
echo (sin(M_PI) . "<br />");
echo (sin(M_PI_2));
?>
```

Hasil:

```
0.14112000806
-0.14112000806
0
1.22460635382E-016
1
```

### 105. cos()

Digunakan untuk menghasilkan nilai *cosine* dari bilangan.

**Syntax dasar:**

```
cos(x)
```

Contoh:

```
<?php
echo(cos(3) . "<br />");
echo(cos(-3) . "<br />");
echo(cos(0) . "<br />");
echo(cos(M_PI) . "<br />");
echo(cos(2*M_PI)
?>
```

Hasil:

```
-0.9899924966004454
-0.9899924966004454
1
-1
1
```

### 106. tan()

Digunakan untuk menghasilkan nilai *tangent* dari bilangan sudut.

**Syntax dasar:**

```
tan(x)
```

Contoh:

```
<?php
echo(tan(M_PI_4) . "<br />");
```

```
echo(tan(0.50) . "<br />");
echo(tan(-0.50) . "<br />");
echo(tan(5) . "<br />");
echo(tan(10) . "<br />");
echo(tan(-5) . "<br />");
echo(tan(-10))
?>
```

Hasil:

```
1
0.546302489844
-0.546302489844
-3.38051500625
0.648360827459
3.38051500625
-0.648360827459
```

### 107. asin()

Digunakan untuk menghasilkan nilai *arcsine* dari bilangan di antara  $-\pi/2$  dan  $\pi/2$  radian..

**Syntax dasar:**

```
asin(x) //nilai x antara -1 dan 1
```

Contoh:

```
<?php
echo(asin(0.64) . "<br />");
echo(asin(0) . "<br />");
echo(asin(-1) . "<br />");
echo(asin(1) . "<br />");
echo(asin(2))
?>
```

Hasil:

```
0.694498265627
0
-1.57079632679
1.57079632679
-1.#IND
```

**Keterangan:**

Jika nilai bilangan di luar jangkauan, maka akan menghasilkan nilai **-1.#IND**. Jika nilai bilangan **1**, maka akan menghasilkan nilai **PI/2**.

**108. acos()**

Digunakan untuk menghasilkan nilai *arccosine* dari bilangan di antara 0 dan PI radian.

**Syntax dasar:**

`asin(x) //nilai x antara -1 dan 1`

**Contoh:**

```
<?php
echo(acos(0.64) . "<br />");
echo(acos(0) . "<br />");
echo(acos(-1) . "<br />");
echo(acos(1) . "<br />");
echo(acos(2))
?>
```

**Hasil:**

```
0.876298061168
1.57079632679
3.14159265359
0
-1.#IND
```

**Keterangan:**

Jika nilai bilangan di luar jangkauan, maka akan menghasilkan nilai **-1.#IND**. Jika nilai bilangan **-1**, maka akan menghasilkan nilai **PI**.

**109. atan()**

Digunakan untuk menghasilkan nilai *arctangent* dari bilangan di antara  $-PI/2$  dan  $PI/2$  radian.

**Syntax dasar:**

`atan(x) dan atan2(x,y)`

**Contoh 1:**

```
<?php
echo(atan(0.50) . "<br />");
echo(atan(-0.50) . "<br />");
echo(atan(5) . "<br />");
echo(atan(10) . "<br />");
echo(atan(-5) . "<br />");
echo(atan(-10))
?>
```

**Hasil:**

```
0.463647609001
-0.463647609001
1.37340076695
1.4711276743
-1.37340076695
-1.4711276743
```

**Contoh 2:**

```
<?php
echo(atan2(0.50,0.50) . "<br />");
echo(atan2(-0.50,-0.50) . "<br />");
echo(atan2(5,5) . "<br />");
echo(atan2(10,20) . "<br />");
echo(atan2(-5,-5) . "<br />");
echo(atan2(-10,10))
?>
```

**Hasil:**

```
0.785398163397
-2.35619449019
0.785398163397
0.463647609001
-2.35619449019
-0.785398163397
```

**110. base\_convert()**

Digunakan untuk konversi bilangan dari satu basis ke basis yang lain.

**Syntax dasar:**

```
base_convert(number, frombase, tobase)
```

**Contoh 1:**

```
<?php
$oct = "0031";
$dec = base_convert($oct,8,10);
echo "$oct dalam bilangan octal adalah sama
dengan $dec dalam desimal.";
?>
```

**Hasil:**

0031 dalam bilangan octal adalah sama dengan 25 dalam desimal.

**Contoh 2:**

```
<?php
$oct = "364";
$hex = base_convert($oct,8,16);
echo "$oct dalam bilangan octal adalah sama
dengan $hex dalam heksadesimal.";
?>
```

**Hasil:**

364 dalam bilangan octal adalah sama dengan f4 dalam heksadesimal.

**111. bindec()**

Digunakan untuk konversi bilangan biner ke bilangan desimal.

**Syntax dasar:**

```
bindec(binary_number)
```

**Contoh 1:**

```
<?php
echo bindec("0011") . "<br />";
echo bindec("01") . "<br />";
echo bindec("11000110011") . "<br />";
echo bindec("111");
?>
```

**Hasil:**

```
3
1
1587
7
```

**112. decbin()**

Digunakan untuk konversi bilangan desimal ke bilangan biner.

**Syntax dasar:**

```
decbin(decimal_number)
```

**Contoh:**

```
<?php
echo decbin("3") . "<br />";
echo decbin("1") . "<br />";
echo decbin("1587") . "<br />";
echo decbin("7");
?>
```

**Hasil:**

```
11
1
11000110011
111
```

**113. dechex()**

Digunakan untuk konversi bilangan desimal ke heksadesimal.

**Syntax dasar:**

```
dechex(decimal_number)
```

Contoh:

```
<?php
echo dechex("30") . "<br />";
echo dechex("10") . "<br />";
echo dechex("1587") . "<br />";
echo dechex("70");
?>
```

Hasil:

```
1e
a
633
46
```

**114. decoct()**

Digunakan untuk konversi bilangan desimal ke bilangan oktal.

**Syntax dasar:**

```
decoct(decimal_number)
```

Contoh:

```
<?php
echo decoct("30") . "<br />";
echo decoct("10") . "<br />";
echo decoct("1587") . "<br />";
echo decoct("70");
?>
```

Hasil:

```
36
12
3063
106
```

**115. deg2rad()**

Digunakan untuk konversi bilangan *degree* ke radian.

**Syntax dasar:**

```
deg2rad(degree_number)
```

Contoh:

```
<?php
$deg = 180;
$rad = deg2rad($deg);
echo "$deg degree sama dengan $rad radian";
?>
```

Hasil:

```
180 degree sama dengan 3.14159265359 radian
```

**116. ceil()**

Digunakan untuk pembulatan ke atas suatu bilangan pecahan.

**Syntax dasar:**

```
ceil(x)
```

Contoh:

```
<?php
echo(ceil(0.60) . "<br />");
echo(ceil(0.20) . "<br />");
echo(ceil(4.2) . "<br />");
echo(ceil(4.7) . "<br />");
echo(ceil(-5.9));
?>
```

Hasil:

```
1
1
5
5
-5
```

**117. floor()**

Digunakan untuk pembulatan ke bawah suatu bilangan pecahan.

**Syntax dasar:**

```
floor(x)
```

Contoh:

```
<?php
echo(floor(0.60) . "<br />");
echo(floor(0.40) . "<br />");
echo(floor(5) . "<br />");
echo(floor(5.1) . "<br />");
echo(floor(-5.1) . "<br />");
echo(floor(-5.9))
?>
```

Hasil:

```
0
0
5
5
-6
-6
```

### 118. round()

Digunakan untuk pembulatan bilangan.

**Syntax dasar:**

```
round(x,prec)
```

Contoh:

```
<?php
echo(round(0.60) . "<br />");
echo(round(0.50) . "<br />");
echo(round(0.49) . "<br />");
echo(round(-4.40) . "<br />");
echo(round(-4.60))
?>
```

Hasil:

```
1
1
0
-4
-5
```

### 119. fmod()

Digunakan untuk mendapatkan sisa hasil pembagian bilangan.

**Syntax dasar:**

```
fmod(x,y)
```

Contoh:

```
<?php
$r = fmod(5,2);
echo $r
?>
```

Hasil:

```
1
```

### 120. exp()

Digunakan untuk menyatakan pangkat bilangan E ( $E^x$ ), di mana E merupakan konstanta Euler (2,7183).

**Syntax dasar:**

```
exp(x)
```

Contoh:

```
<?php
echo(exp(1) . "<br />");
echo(exp(-1) . "<br />");
echo(exp(5) . "<br />");
echo(exp(10))
?>
```

Hasil:

```
2.718281828459045
0.36787944117144233
148.4131591025766
22026.465794806718
```

**121. expm1()**

Digunakan untuk menyatakan bilangan  $E^x - 1$ , di mana  $E$  merupakan konstanta Euler (2,7183).

**Syntax dasar:**

`expm1(x)`

**Contoh:**

```
<?php
echo(expm1(1)) . "<br />";
echo(expm1(-1)) . "<br />";
echo(expm1(5)) . "<br />";
echo(expm1(10))
?>
```

**122. log()**

Digunakan untuk operasi logaritma pada suatu bilangan.

**Syntax dasar:**

`Log(x, base)`

**Contoh:**

```
<?php
echo(log(2.7183)) . "<br />";
echo(log(2)) . "<br />";
echo(log(1)) . "<br />";
echo(log(0)) . "<br />";
echo(log(-1))
?>
```

**Hasil:**

```
1.00000668491
0.69314718056
0
-1.#INF
-1.#IND
```

**Keterangan:**

Jika parameter  $x$  bernilai negatif, maka akan menghasilkan **-1.#IND**.

**123. log10()**

Digunakan untuk operasi logaritma basis 10 pada suatu bilangan.

**Syntax dasar:**

`Log10(x)`

**Contoh:**

```
<?php
echo(log10(2.7183)) . "<br />";
echo(log10(2)) . "<br />";
echo(log10(1)) . "<br />";
echo(log10(0)) . "<br />";
echo(log10(-1))
?>
```

**Hasil:**

```
0.434297385125
0.301029995664
0
-1.#INF
-1.#IND
```

**Keterangan:**

Jika parameter  $x$  bernilai negatif, maka akan menghasilkan **-1.#IND**.

**124. max()**

Digunakan untuk mendapatkan nilai terbesar dari perbandingan dua buah bilangan.

**Syntax dasar:**

`max(x, y)`

**Contoh:**

```
<?php
echo(max(5,7)) . "<br />";
echo(max(-3,5)) . "<br />";
echo(max(-3,-5)) . "<br />";
echo(max(7.25,7.30))
?>
```

Hasil:

```
7
5
-3
7.3
```

### 125. min()

Digunakan untuk mendapatkan nilai terkecil dari perbandingan dua buah bilangan.

**Syntax dasar:**

```
min(x, y)
```

Contoh:

```
<?php
echo(min(5,7) . "<br />");
echo(min(-3,5) . "<br />");
echo(min(-3,-5) . "<br />");
echo(min(7.25,7.30))
?>
```

Hasil:

```
5
-3
-5
7.25
```

### 126. sqrt()

Digunakan untuk mendapatkan nilai akar dari suatu bilangan.

**Syntax dasar:**

```
sqrt(x)
```

Contoh:

```
<?php
echo(sqrt(0) . "<br />");
echo(sqrt(1) . "<br />");
echo(sqrt(9) . "<br />");
echo(sqrt(0.64) . "<br />");
echo(sqrt(-9))
?>
```

Hasil:

```
0
1
3
0.8
-1.#IND
```

**Keterangan:**

Jika parameter **x** bernilai negatif, maka akan menghasilkan **-1.#IND**.

### 127. rand()

Digunakan untuk menghasilkan nilai acak dari bilangan integer.

**Syntax dasar:**

```
rand(min, max)
```

Contoh:

```
<?php
echo(rand() . "<br />");
echo(rand() . "<br />");
echo(rand(10,100))
?>
```

Hasil:

```
17757
3794
97
```

### 128. pow()

Digunakan untuk menghasilkan nilai pangkat bilangan.

**Syntax dasar:**

```
pow(x, y) // berarti x pangkat y
```

Contoh:

```
<?php
echo pow(4,2) . "<br />";
```

```
echo pow(6,2) . "<br />";
echo pow(-6,2) . "<br />";
echo pow(-6,-2);
?>
```

Hasil:

```
16
36
36
0.02777777777778
```

### 129. pi()

Digunakan untuk menghasilkan nilai PI.

**Syntax dasar:**

```
pi()
```

Contoh:

```
<?php
echo pi();
?>
```

Hasil:

```
3.14159265359
```

### 130. echo()

Merupakan fungsi yang digunakan untuk menghasilkan keluaran satu atau lebih string.

**Syntax dasar:**

```
echo(strings)
```

Contoh:

```
<?php
$str = "PHP adalah server side scripting";
echo $str;
echo "<br />";
echo $str."<br />HTML merupakan kode dasar
membuat web";
?>
```

Hasil:

```
PHP adalah server side scripting
PHP adalah server side scripting
HTML merupakan kode dasar membuat web
```

### 131. strlen()

Merupakan fungsi yang digunakan untuk menghasilkan nilai dari panjang sebuah *string*.

Contoh:

```
<?php
echo strlen("PHP adalah server side script");
?>
```

Hasil: 29

**Keterangan:**

Setiap karakter termasuk spasi akan dihitung.

### 132. strpos()

Merupakan fungsi yang digunakan untuk mencari lokasi *string* tertentu dalam sebuah teks..

Contoh:

```
<?php
echo strpos("PHP adalah server side script",
"server");
?>
```

Hasil: 11

**Keterangan:**

Hitungan dimulai dari 0. Posisi *string* "server" pada contoh di atas dimulai dari urutan ke-11 pada teks.

### 133. chr()

Merupakan fungsi yang digunakan untuk menghasilkan karakter dari nilai ASCII tertentu.

**Syntax dasar:**

```
chr(ascii)
```

Contoh:

```
<?php
echo chr(52) . "<br />";
echo chr(052) . "<br />";
echo chr(0x52) . "<br />";
?>
```

Hasil:

```
4
*
R
```

### 134. trim()

Merupakan fungsi yang digunakan untuk menghapus karakter spasi dan beberapa karakter lain dari kedua sisi ujung *string*.

**Syntax dasar:**

```
trim(string, charlist)
```

**Keterangan:**

Parameter **charlist** bersifat pilihan. Jika tidak diisi, maka fungsi ini akan memeriksa beberapa karakter, yaitu:

- `"\0"` - ASCII 0, NULL.
- `"\t"` - ASCII 9, karakter Tab.
- `"\n"` - ASCII 10, karakter ganti baris.
- `"\x0B"` - ASCII 11, karakter Tab vertikal.
- `"\r"` - ASCII 13, karakter Enter.
- `" "` - ASCII 32, karakter spasi.

Contoh:

```
<?php
$str = "\r\nHello World!\r\n";
echo "Tanpa trim: " . $str;
echo "<br />";
echo "Menggunakan trim: " . trim($str);
?>
```

### 135. chop() atau rtrim()

Merupakan fungsi yang digunakan untuk menghapus karakter spasi dan beberapa karakter lain dari ujung kanan *string*. Fungsi ini merupakan alternatif dari fungsi **rtrim()**.

**Syntax dasar:**

```
chop(string, charlist)
```

**Keterangan:**

Parameter **charlist** bersifat pilihan. Jika tidak diisi, maka fungsi ini akan memeriksa beberapa karakter, yaitu:

- `"\0"` - ASCII 0, NULL.
- `"\t"` - ASCII 9, karakter Tab.
- `"\n"` - ASCII 10, karakter ganti baris.
- `"\x0B"` - ASCII 11, karakter Tab vertikal.
- `"\r"` - ASCII 13, karakter Enter.
- `" "` - ASCII 32, karakter spasi.

Contoh:

```
<?php
$str = "Hello World!\n\n";
echo $str;
echo chop($str);
?>
```

Hasil:

```
Hello World! Hello World!
```

**Keterangan:**

Fungsi **chop()** pada contoh di atas dapat diganti dengan fungsi **rtrim()**.

### 136. ltrim()

Merupakan fungsi yang digunakan untuk menghapus karakter spasi dan beberapa karakter lain dari ujung kiri *string*.

**Syntax dasar:**

```
ltrim(string, charlist)
```

**Keterangan:**

Parameter **charlist** bersifat pilihan. Jika tidak diisi, maka fungsi ini akan memeriksa beberapa karakter, yaitu:

- "0" - ASCII 0, NULL.
- "t" - ASCII 9, karakter Tab.
- "n" - ASCII 10, karakter ganti baris.
- "x0B" - ASCII 11, karakter Tab vertikal.
- "r" - ASCII 13, karakter Enter.
- " " - ASCII 32, karakter spasi.

**Contoh:**

```
<?php
$str = "    Hello World!";
echo "Tanpa ltrim: " . $str;
echo "<br/>";
echo "Menggunakan ltrim: " . ltrim($str);
?>
```

**137. print()**

Digunakan untuk menampilkan satu atau lebih *string*.

**Syntax dasar:**

```
print(strings)
```

**Contoh:**

```
<?php
$str = "PHP";
print $str;
print "<br/>";
print $str."<br/>digunakan untuk membuat web dinamis";
?>
```

**Hasil:**

```
PHP
PHP
digunakan untuk membuat web dinamis
```

**138. fprintf()**

Digunakan untuk memformat *string*.

**Syntax dasar:**

```
fprintf(stream, format, arg1, arg2, arg++)
```

**Keterangan:**

Parameter format:

- %% - Menghasilkan tanda persen.
- %b - Bilangan biner.
- %c - Karakter sesuai dengan nilai ASCII.
- %d - Bilangan desimal.
- %e - Notasi ilmiah, seperti 1.2e+2.
- %u - Bilangan desimal (*unsigned*).
- %f - Bilangan pecahan (tergantung dari seting lokal).
- %F - Bilangan pecahan (tidak tergantung dari seting lokal).
- %o - Bilangan oktal.
- %s - *String*.
- %x - Bilangan heksadesimal (huruf kecil).
- %X - Bilangan heksadesimal (huruf besar).

**Contoh:**

```
<?php
$bil = 123;
$file = fopen("catatan.txt", "w");
fprintf($file, "dengan 2 desimal: %1\$.2f
Tanpa desimal: %1\$.0f", $bil);
?>
```

**Hasil:**

```
Dengan 2 desimal: 123.00
Tanpa desimal: 123
```

**139. explode()**

Digunakan untuk memasukkan *string* ke dalam bentuk *array* berdasarkan tanda pemisah tertentu.

**Syntax dasar:**

```
explode(separator, string, limit)
```

Contoh:

```
<?php
$str = "Life is beautiful";
print_r (explode(" ", $str));
?>
```

Hasil:

```
Array ([0] => Life [1] => is [2] => beautiful)
```

#### 140. money\_format()

Digunakan untuk memformat *string* sebagai mata uang

**Syntax dasar:**

```
money_format(string, number)
```

Contoh:

```
<?php
$number = 1234.56;
setlocale(LC_MONETARY, "en_US");
echo money_format("Harga dalam USD %i", $number);
?>
```

Hasil:

```
Harga dalam USD 1,234.56
```

#### 141. number\_format()

Digunakan untuk memformat nomor dalam grup ribuan.

**Syntax dasar:**

```
number_format(number, decimals, decimalpoint,
separator)
```

Contoh:

```
<?php
echo number_format("1000000");
echo "<br />";
echo number_format("1000000", 2);
echo "<br />";
echo number_format("1000000", 2, ",", ".");
?>
```

Hasil:

```
1,000,000
1,000,000.00
1.000.000,00
```

#### 142. nl2br()

Digunakan untuk menambahkan tag HTML *line breaks* (<br>) setelah tanda *newline* (\n) dalam *string*.

**Syntax dasar:**

```
nl2br(string)
```

Contoh:

```
<?php
echo nl2br("PHP\nServer side scripting.");
?>
```

Hasil:

```
PHP
Server side scripting.
```

#### 143. ord()

Digunakan untuk menghasilkan nilai ASCII dari karakter pertama sebuah *string*.

**Syntax dasar:**

```
ord(string)
```

Contoh:

```
<?php
echo ord("h") . "<br />";
echo ord("hello") . "<br />";
?>
```

Hasil:

```
104
104
```

### 144. substr()

Digunakan untuk mendapatkan bagian tertentu dari sebuah *string*.

#### Syntax dasar:

```
substr(string, start, length)
```

#### Contoh:

```
<?php
echo substr("Hello world!", 6, 5);
?>
```

#### Hasil:

```
world
```

#### Keterangan:

Perintah **substr()** pada contoh di atas akan dimulai dari karakter ke-6 dan sebanyak 5 karakter.

### 145. substr\_replace()

Digunakan untuk mengisi bagian *string* tertentu dengan *string* lain.

#### Syntax dasar:

```
substr_replace(string, replacement, start, length)
```

#### Contoh:

```
<?php
echo substr_replace("HTML kode dasar membuat
web", "website", 24);
?>
```

#### Hasil:

```
HTML kode dasar membuat website
```

#### Keterangan:

Perintah **substr\_replace()** pada contoh di atas akan mengisi teks "web" yang berada di urutan karakter ke-24 dengan *string* "website".

# 3

## Syntax SQL (Structure Query Language)

### 1. Create Database

Digunakan untuk membuat *database* baru.

#### Syntax dasar:

```
CREATE DATABASE database_name
```

#### Contoh:

```
CREATE DATABASE databasaku
```

### 2. Create Table

Digunakan untuk membuat tabel data baru dalam sebuah *database*.

#### Syntax dasar:

```
CREATE TABLE table_name
(
    column_name1 data_type,
    column_name2 data_type,
    column_name3 data_type,
    ....
)
```

#### Contoh:

```
CREATE TABLE bukutamu
(
    id int,
    nama varchar(255),
    email varchar(50),
    kota varchar(255)
)
```

### 3. Select

Digunakan untuk memilih data dari tabel *database*.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name
Atau
SELECT * FROM table_name
```

Contoh 1:

```
SELECT nama,email FROM bukutamu
```

Contoh 2:

```
SELECT * FROM bukutamu
```

### 4. Select Distinct

Digunakan untuk memilih data-data yang berbeda (menghilangkan duplikasi) dari sebuah table database.

#### Syntax dasar:

```
SELECT DISTINCT column_name(s)
FROM table_name
```

Contoh:

```
SELECT DISTINCT kota FROM bukutamu
```

### 5. Where

Digunakan untuk memfilter data pada perintah **Select**.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

Contoh:

```
SELECT * FROM bukutamu
WHERE kota='YOGYAKARTA'
```

### 6. Order By

Digunakan untuk mengurutkan data berdasarkan kolom (*field*) tertentu. Secara *default*, urutan tersusun secara *ascending* (urut kecil ke besar). Anda dapat mengubahnya menjadi *descending* (urut besar ke kecil) dengan menambahkan perintah **DESC**.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

Contoh 1:

```
SELECT * FROM bukutamu
ORDER BY nama
```

Contoh 2:

```
SELECT * FROM bukutamu
ORDER BY id DESC
```

### 7. Like

Digunakan bersama dengan perintah **Where**, untuk proses pencarian data dengan spesifikasi tertentu.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name
WHERE column_name LIKE pattern
```

Contoh 1:

```
SELECT * FROM bukutamu
WHERE nama LIKE 'a%'
```

#### Keterangan:

Contoh di atas digunakan untuk pencarian berdasarkan kolom nama yang berhuruf depan "a".

Contoh 2:

```
SELECT * FROM bukutamu
WHERE nama LIKE '%a'
```

**Keterangan:**

Contoh di atas digunakan untuk pencarian berdasarkan kolom nama yang berhuruf belakang "a".

**8. In**

Digunakan untuk pencarian data menggunakan lebih dari satu filter pada perintah **Where**.

**Syntax dasar:**

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1,value2,...)
```

**Contoh:**

```
SELECT * FROM buku_tamu
WHERE kota IN ('Yogyakarta','Jakarta')
```

**9. Between**

Digunakan untuk menentukan jangkauan pencarian.

**Syntax dasar:**

```
SELECT column_name(s)
FROM table_name
WHERE column_name
BETWEEN value1 AND value2
```

**Contoh:**

```
SELECT * FROM buku_tamu
WHERE id
BETWEEN 5 AND 15
```

**Keterangan:**

Contoh di atas digunakan untuk mencari data yang memiliki nomor id antara 5 dan 15.

**10. Insert Into**

Digunakan untuk menambahkan data baru di tabel *database*.

**Syntax dasar:**

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

**Atau**

```
INSERT INTO table_name (column1, column2,
column3,...)
VALUES (value1, value2, value3,...)
```

**Contoh 1:**

```
INSERT INTO buku_tamu
VALUES (1,'Arini', 'arini@mail.com', 'Yogya-
karta')
```

**Contoh 2:**

```
INSERT INTO buku_tamu (id, nama, email, kota)
VALUES (1, 'Arini', 'arini@mail.com', 'Yog-
yakarta')
```

**11. Update**

Digunakan untuk mengubah/memperbarui data di tabel *database*.

**Syntax dasar:**

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

**Contoh:**

```
UPDATE buku_tamu
SET email='arini@yahoo.com', kota='Jakarta'
WHERE id=1
```

**12. Delete**

Digunakan untuk menghapus data di tabel *database*. Tambahkan perintah **Where** untuk memfilter data-data tertentu yang akan dihapus. Jika tanpa perintah **Where**, maka seluruh data dalam tabel akan terhapus.

**Syntax dasar:**

```
DELETE FROM table_name
WHERE some_column=some_value
```

**Contoh:**

```
DELETE FROM buku_tamu
WHERE id=1
```

### 13. Inner Join atau Join

Digunakan untuk menghasilkan baris data dengan cara menggabungkan 2 buah tabel atau lebih menggunakan pasangan data yang *match* pada masing-masing tabel. Perintah ini sama dengan perintah **Join** yang sering digunakan.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name1
INNER JOIN table_name2
ON table_name1.column_name=table_name2.
column_name
```

#### Contoh:

```
SELECT bukutamu.nama, bukutamu.emai,
       order.no_order
FROM bukutamu
INNER JOIN order
ON bukutamu.id=order.id
ORDER BY bukutamu.nama
```

### 14. Left Join

Digunakan untuk menghasilkan baris data dari tabel kiri (nama tabel pertama) yang tidak ada pasangan datanya pada tabel kanan (nama tabel kedua).

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name1
LEFT JOIN table_name2
ON table_name1.column_name=table_name2.
column_name
```

#### Contoh:

```
SELECT bukutamu.nama, bukutamu.emai,
       order.no_order
FROM bukutamu
LEFT JOIN order
ON bukutamu.id=order.id
ORDER BY bukutamu.nama
```

### 15. Right Join

Digunakan untuk menghasilkan baris data dari tabel kanan (nama tabel kedua) yang tidak ada pasangan datanya pada tabel kiri (nama tabel pertama).

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name1
RIGHT JOIN table_name2
ON table_name1.column_name=table_name2.
column_name
```

#### Contoh:

```
SELECT bukutamu.nama, bukutamu.email,
       order.no_order
FROM bukutamu
RIGHT JOIN order
ON bukutamu.id=order.id
ORDER BY bukutamu.nama
```

### 16. Full Join

Digunakan untuk menghasilkan baris data jika ada data yang sama pada salah satu tabel.

#### Syntax dasar:

```
SELECT column_name(s)
FROM table_name1
FULL JOIN table_name2
ON table_name1.column_name=table_name2.
column_name
```

#### Contoh:

```
SELECT bukutamu.nama, bukutamu.email,
       order.no_order
FROM bukutamu
FULL JOIN order
ON bukutamu.id=order.id
ORDER BY bukutamu.nama
```

## 17. Union

Digunakan untuk menggabungkan hasil dari 2 atau lebih perintah **Select**.

### Syntax dasar:

```
SELECT column_name(s) FROM table_name1
UNION
SELECT column_name(s) FROM table_name2
Atau
SELECT column_name(s) FROM table_name1
UNION ALL
SELECT column_name(s) FROM table_name2
```

### Contoh:

```
SELECT nama FROM mhs_kampus1
UNION
SELECT nama FROM mhs_kampus2
```

## 18. Alter Table

Digunakan untuk menambah, hapus, atau mengubah kolom (*field*) pada tabel yang sudah ada.

### Syntax untuk menambah kolom:

```
ALTER TABLE table_name
ADD column_name datatype
```

### Contoh:

```
ALTER TABLE Persons
ADD DateOfBirth date
```

### Syntax untuk menghapus kolom:

```
ALTER TABLE table_name
DROP COLUMN column_name
```

### Contoh:

```
ALTER TABLE Persons
DROP COLUMN DateOfBirth
```

### Syntax untuk mengubah kolom:

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype
```

### Contoh:

```
ALTER TABLE Persons
ALTER COLUMN DateOfBirth year
```

## 19. Now()

Digunakan untuk mendapatkan informasi waktu (tanggal dan jam) saat ini.

### Syntax dasar:

```
Now()
```

### Contoh:

```
SELECT NOW()
```

## 20. Curdate()

Digunakan untuk mendapatkan informasi tanggal saat ini.

### Syntax dasar:

```
Curdate()
```

### Contoh:

```
SELECT CURDATE()
```

## 21. Curtime()

Digunakan untuk mendapatkan informasi jam saat ini.

### Syntax dasar:

```
Curtime()
```

### Contoh:

```
SELECT CURTIME()
```

## 22. Extract()

Digunakan untuk mendapatkan informasi bagian-bagian dari data waktu tertentu, seperti tahun, bulan, hari, jam, menit, dan detik tertentu.

### Syntax dasar:

```
Extract(unit FROM date)
```

**Keterangan:**

Parameter unit dapat berupa:

- MICROSECOND
- SECOND
- MINUTE
- HOUR
- DAY
- WEEK
- MONTH
- QUARTER
- YEAR
- SECOND\_MICROSECOND
- MINUTE\_MICROSECOND
- MINUTE\_SECOND
- HOUR\_MICROSECOND
- HOUR\_SECOND
- HOUR\_MINUTE
- DAY\_MICROSECOND
- DAY\_SECOND
- DAY\_MINUTE
- DAY\_HOUR
- YEAR\_MONTH

Contoh:

```
SELECT EXTRACT(YEAR FROM tglorder) AS Th_Order,
EXTRACT(MONTH FROM tglorder) AS Bulan_Order,
EXTRACT(DAY FROM tglorder) AS Hari_Order,
FROM order
WHERE id=1
```

**23. Date\_Add() dan Date\_Sub()**

Fungsi **Date\_Add()** digunakan untuk menambahkan interval waktu tertentu pada sebuah tanggal, sedangkan fungsi **Date\_Sub()** digunakan untuk pengurangan sebuah tanggal dengan interval waktu tertentu.

**Syntax dasar:**

```
DATE_ADD(date, INTERVAL expr type)
DATE_SUB(date, INTERVAL expr type)
```

**Keterangan:**

Tipe pada parameter **INTERVAL** dapat berupa:

- MICROSECOND
- SECOND
- MINUTE
- HOUR
- DAY
- WEEK
- MONTH
- QUARTER
- YEAR
- SECOND\_MICROSECOND
- MINUTE\_MICROSECOND
- MINUTE\_SECOND
- HOUR\_MICROSECOND
- HOUR\_SECOND
- HOUR\_MINUTE
- DAY\_MICROSECOND
- DAY\_SECOND
- DAY\_MINUTE
- DAY\_HOUR
- YEAR\_MONTH

Contoh 1:

```
SELECT id, DATE_ADD(tglorder, INTERVAL 30 DAY)
AS Waktu_pembayaran
FROM order
```

Contoh 2:

```
SELECT id, DATE_SUB(tglorder, INTERVAL 5 DAY)
AS Pengurangan_Waktu
FROM order
```

**24. DateDiff()**

Digunakan untuk mendapatkan informasi waktu di antara 2 buah tanggal.

**Syntax dasar:**

```
DATEDIFF(date1, date2)
```

Contoh:

```
SELECT DATEDIFF('2010-06-30', '2010-06-29')
AS Selisih_waktu
```

**25. Date\_Format()**

Digunakan untuk menampilkan informasi jam dan tanggal dengan format tertentu.

**Syntax dasar:**

```
DATE_FORMAT(date, format)
```

**Keterangan:**

Parameter format dapat berupa:

- %a, nama hari yang disingkat
- %b, nama bulan yang disingkat
- %c, bulan (numerik)
- %D, hari dalam sebulan dengan format English
- %d, hari dalam sebulan (numerik 00-31)
- %e, hari dalam sebulan (numerik 0-31)
- %f, micro detik
- %H, jam (00-23)
- %h, jam (01-12)
- %I, jam (01-12)
- %i, menit (00-59)
- %j, hari dalam setahun (001-366)

- %k, jam (0-23)
- %l, jam (1-12)
- %M, nama bulan
- %m, bulan (numerik 00-12)
- %p, AM atau PM
- %r, waktu jam dalam format 12 jam (hh:mm:ss AM or PM)
- %S, detik (00-59)
- %s, detik (00-59)
- %T, waktu jam dalam format 24 jam (hh:mm:ss)
- %U, minggu (00-53) di mana Sunday sebagai hari pertama dalam seminggu
- %u, minggu (00-53) di mana Monday sebagai hari pertama dalam seminggu
- %W, nama hari kerja
- %w, hari dalam seminggu (0=Sunday, 6=Saturday)
- %X, tahun dalam seminggu di mana Sunday sebagai hari pertama dalam seminggu (4 digits) digunakan dengan %V
- %x, tahun dalam seminggu di mana Monday sebagai hari pertama dalam seminggu (4 digits) digunakan dengan %v
- %Y, tahun 4 digit
- %y, tahun 2 digit

Contoh:

```
DATE_FORMAT(NOW(), '%b %d %Y %h:%i %p')
DATE_FORMAT(NOW(), '%m-%d-%Y')
DATE_FORMAT(NOW(), '%d %b %y')
DATE_FORMAT(NOW(), '%d %b %Y %T:%f')
```

Hasil:

```
Nov 04 2009 11:45 PM
11-04-2009
04 Nov 09
04 Nov 2009 11:45:34.243
```

## 26. Drop Table()

Digunakan untuk menghapus tabel beserta seluruh datanya.

**Syntax dasar:**

```
DROP TABLE table_name
```

Contoh:

```
DROP TABLE mhs
```

## 27. Drop Database()

Digunakan untuk menghapus database.

**Syntax:**

```
DROP DATABASE database_name
```

Contoh:

```
DROP DATABASE my_db
```

## 28. AVG()

Digunakan untuk menghitung nilai rata-rata dari suatu data.

**Syntax dasar:**

```
SELECT AVG(column_name) FROM table_name
```

Contoh:

```
SELECT AVG(harga) AS Harga_rata2 FROM order
```

## 29. Count()

Digunakan untuk menghitung jumlah (cacah) suatu data.

**Syntax dasar:**

```
SELECT COUNT(column_name) FROM table_name
```

Contoh:

```
SELECT COUNT(id) AS Jumlah_tamu FROM bukutamu
```

## 30. Max()

Digunakan untuk mendapatkan nilai terbesar dari data-data yang ada.

**Syntax dasar:**

```
SELECT MAX(column_name) FROM table_name
```

Contoh:

```
SELECT MAX(harga) AS Harga_termahal FROM order
```

**31. Min()**

Digunakan untuk mendapatkan nilai terkecil dari data-data yang ada.

**Syntax dasar:**

```
SELECT MIN(column_name) FROM table_name
```

**Contoh:**

```
SELECT MIN(harga) AS Harga_termurah FROM order
```

**32. Sum()**

Digunakan untuk mendapatkan nilai total penjumlahan dari data-data yang ada.

**Syntax dasar:**

```
SELECT SUM(column_name) FROM table_name
```

**Contoh:**

```
SELECT SUM(harga) AS Harga_total FROM order
```

**33. Group By()**

Digunakan untuk mengelompokkan data dengan kriteria tertentu.

**Syntax dasar:**

```
SELECT column_name, aggregate_
      function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name
```

**Contoh:**

```
SELECT nama_customer, SUM(harga) FROM order
GROUP BY nama_customer
```

**34. Having()**

Digunakan untuk memfilter data dengan fungsi tertentu.

**Syntax dasar:**

```
SELECT column_name, aggregate_
      function(column_name)
FROM table_name
```

```
WHERE column_name operator value
GROUP BY column_name
HAVING aggregate_function(column_name)
      operator value
```

**Contoh:**

```
SELECT nama_customer, SUM(harga) FROM order
WHERE nama_customer='Arini' OR
      nama_customer='Maheswari'
GROUP BY nama_customer
HAVING SUM(harga)>25000
```

**35. Ucase()**

Digunakan untuk mengubah huruf pada data tertentu menjadi huruf besar.

**Syntax dasar:**

```
SELECT UCASE(column_name) FROM table_name
```

**Contoh:**

```
SELECT UCASE(nama) as Nama FROM bukutamu
```

**36. Lcase()**

Digunakan untuk mengubah huruf pada data tertentu menjadi huruf kecil.

**Syntax dasar:**

```
SELECT LCASE(column_name) FROM table_name
```

**Contoh:**

```
SELECT LCASE(nama) as Nama FROM bukutamu
```

**37. Mid()**

Digunakan untuk mengambil beberapa karakter dari field teks.

**Syntax dasar:**

```
SELECT MID(column_name, start[, length]) FROM
table_name
```

## Syntax SQL

Contoh:

```
SELECT MID(kota,1,4) as singkatan_kota FROM
bukutamu
```

### 38. Len()

Digunakan untuk mendapatkan informasi jumlah karakter dari *field* teks.

**Syntax dasar:**

```
SELECT LEN(column_name) FROM table_name
```

Contoh:

```
SELECT LEN(nama) as panjang_nama
FROM bukutamu
```

### 39. Round()

Digunakan untuk pembulatan bilangan pecahan.

**Syntax dasar:**

```
SELECT ROUND(column_name,decimals)
FROM table_name
```

Contoh:

```
SELECT no_mhs, ROUND(nilai,0) as nilai_bulat
FROM tnilai
```

# 4

## Syntax Visual Basic

### 1. MsgBox

Digunakan untuk menampilkan pesan pada kotak dialog.

**Syntax dasar:**

```
MsgBox(prompt[, buttons] [, title]
[, helpfile, context])
```

**Keterangan:**

Parameter **buttons** dapat diisi dengan:

**vbOKOnly**, menampilkan hanya tombol **OK**.

**vbOKCancel**, menampilkan tombol **OK** dan **Cancel**.

**vbAbortRetryIgnore**, menampilkan tombol **Abort**, **Retry**, **Ignore**.

**vbYesNoCancel**, menampilkan tombol **Yes**, **No**, **Cancel**.

**vbYesNo**, menampilkan tombol **Yes** dan **No**.

**vbRetryCancel**, menampilkan tombol **Retry** dan **Cancel**.

**Ikon parameter buttons:**

|            |                                                                                     |               |                                                                                     |
|------------|-------------------------------------------------------------------------------------|---------------|-------------------------------------------------------------------------------------|
| vbCritical |  | vbExclamation |  |
| vbQuestion |  | vbInformation |  |

Contoh:

```
Private Sub Command1_Click()
MsgBox "Kolom email harus diisi ", _
vbExclamation, "Peringatan"
End Sub
```

Hasil:



## 2. If... Then... Else...

Digunakan sebagai statemen kondisi. Sebuah kode/blok kode akan dijalankan apabila kondisi terpenuhi (*true*), dan blok kode lainnya akan dijalankan apabila kondisi tidak terpenuhi (*false*).

### Syntax 1:

```
If condition1 Then statement
```

### Contoh:

```
If nilai_angka >= 80 Then nilai_huruf = "A"
```

### Syntax 2:

```
If condition1 Then
    statement
End If
```

### Contoh:

```
If nilai_angka >= 80 Then
    nilai_huruf = "A"
    MsgBox "Selamat Anda mendapat nilai terbaik", _
        vbInformation, "Informasi"
End If
```

### Syntax 3:

```
If condition1 Then
    statement
Else
    statement
End If
```

### Contoh:

```
If nilai_angka >= 80 Then
    nilai_huruf = "A"
    MsgBox "Selamat Anda mendapat nilai terbaik", _
        vbInformation, "Informasi"
Else
    MsgBox "Anda harus belajar lebih giat lagi", _
        vbInformation, "Informasi"
End If
```

### Syntax 4:

```
If condition1 Then
    statement
ElseIf condition2 Then
    statement
Else
    statement
End If
```

### Contoh:

```
If nilai_angka >= 80 Then
    nilai_huruf = "A"
    MsgBox "Selamat Anda mendapat nilai terbaik", _
        vbInformation, "Informasi"
ElseIf nilai_angka >= 70 Then
    nilai_huruf = "B"
    MsgBox "Nilai Anda baik", _
        vbInformation, "Informasi"
Else
    MsgBox "Nilai Anda kurang", _
        vbInformation, "Informasi"
End If
```

## 3. Do... Loops

Digunakan sebagai statemen perulangan.

### Syntax 1:

```
Do Until (Expression)
    (Code to execute)
Loop
```

### Contoh:

```
Dim i As Integer
i = 0
Do Until i = 100
    i = i + 1
Loop
```

**Syntax 2:**

```
Do While (Expression)
    (Code to execute)
Loop
```

**Contoh:**

```
Dim i As Integer
Do While i <= 100
    i = i + 1
Loop
```

**Syntax 3:**

```
Do
    (Code to execute)
Loop While (Expression)
```

**Contoh:**

```
Dim i As Integer
Do
    i = i + 1
Loop While i <= 100
```

**4. For... Next**

Digunakan sebagai statemen perulangan.

**Syntax dasar:**

```
For counter = start To end [ Step step ]
    (Code to execute)
Next [ counter ]
```

**Contoh 1:**

```
Dim i As Integer
For i = 0 To 10
    MsgBox "Nilai i = " & i
Next i
```

**Contoh 2:**

```
Dim i As Integer
For i = 0 To 10 step 2
    MsgBox "Nilai i = " & i
Next i
```

**5. Select Case**

Digunakan sebagai statemen kodisi dengan banyak pilihan.

**Syntax dasar:**

```
Select Case <test expression>
    Case <expression list 1>
        <statement list 1>
    Case <expression list 2>
        <statement list 2>
    ...
    Case Else
        <statement list n>
End Select
```

**Contoh:**

```
Select Case nilai_angka
    Case 0 To 50
        nilai_huruf = "D"
    Case Is <= 70
        nilai_huruf = "C"
    Case Is <= 80
        nilai_huruf = "B"
    Case Else
        nilai_huruf = "A"
```

**End Select**

```
MsgBox "Nilai Huruf = " & nilai_huruf
```

**6. Len()**

Digunakan untuk mendapatkan informasi panjang sebuah string.

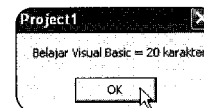
**Syntax dasar:**

```
Len(string)
```

**Contoh:**

```
Dim str As String
str = "Belajar Visual Basic"
MsgBox str & " = " & Len(str) & " karakter"
```

Hasil:



## 7. Mid\$()

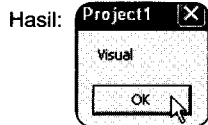
Digunakan untuk mendapatkan bagian tertentu dari sebuah *string*.

### Syntax dasar:

```
Mid$(string, start[, length])
```

### Contoh:

```
Dim str As String
str = Mid$("Belajar Visual Basic", 9, 6)
MsgBox str
```



## 8. Left\$()

Digunakan untuk mendapatkan bagian tertentu dari sebuah *string*, yang dimulai dari sisi kiri *string*.

### Syntax dasar:

```
Left$(string, length)
```

### Contoh:

```
Dim str As String
str = Left$("Belajar Visual Basic", 14)
MsgBox str
```



## 9. Right\$()

Digunakan untuk mendapatkan bagian tertentu dari sebuah *string*, yang dimulai dari sisi kanan *string*.

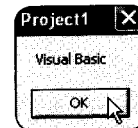
### Syntax dasar:

```
Right$(string, length)
```

### Contoh:

```
Dim str As String
str = Right$("Belajar Visual Basic", 12)
MsgBox str
```

### Hasil:



## 10. UCase\$()

Digunakan untuk mengganti semua huruf pada sebuah teks/*string* menjadi huruf besar (kapital).

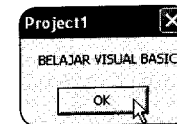
### Syntax dasar:

```
UCase$(string)
```

### Contoh:

```
Dim str As String
str = UCase$("Belajar Visual Basic")
MsgBox str
```

### Hasil:



## 11. LCase\$()

Digunakan untuk mengganti semua huruf pada sebuah teks/*string* menjadi huruf kecil.

### Syntax dasar:

```
LCase$(string)
```

### Contoh:

```
Dim str As String
str = LCase$("Belajar Visual Basic")
MsgBox str
```

Hasil:



## 12. Instr()

Digunakan untuk mencari posisi/letak karakter tertentu dalam sebuah string.

### Syntax dasar:

```
InStr([start,] string1, string2 [, compare])
```

### Contoh:

```
Dim str As String
str = Instr(5, "Belajar Visual Basic", "a")
MsgBox str
```

### Keterangan:

Pada contoh di atas akan dicari posisi karakter "a" yang dihitung mulai dari karakter ke 5.

## 13. String\$()

Digunakan untuk menghasilkan sebuah *string* dari perulangan karakter tertentu dengan panjang yang sudah ditentukan.

### Syntax dasar:

```
String$(number, character)
```

### Contoh:

```
Dim str As String
str = String$(6, "X")
MsgBox str
```

### Keterangan:

Contoh di atas akan menghasilkan sebuah *string* yang terdiri dari 6 buah karakter X.

## 14. Space\$()

Digunakan untuk menghasilkan sebuah *string* dari perulangan karakter spasi dengan panjang yang sudah ditentukan.

### Syntax dasar:

```
Space$(number)
```

### Contoh:

```
Dim str As String
str = Space$(6)
MsgBox str
```

### Keterangan:

Pada contoh di atas akan menghasilkan sebuah *string* yang terdiri dari 6 buah karakter spasi.

## 15. Replace\$()

Digunakan untuk mengganti *substring* atau karakter dalam sebuah *string* dengan *substring* atau karakter yang lain.

### Syntax dasar:

```
Replace$(expression, find,
    replacewith[, start[, count[, compare]]])
```

### Contoh:

```
Dim str As String
str = Replace$("Visual Basic", "Basic", _
    "Studio")
MsgBox str
```

### Keterangan:

Pada contoh di atas, *substring* "Basic" diganti dengan *substring* "Studio", sehingga akan menghasilkan *string* "Visual Studio".

## 16. StrReverse\$()

Digunakan untuk membalik sebuah *string* per karakter.

### Syntax dasar:

```
StrReverse$(string)
```

Contoh:

```
Dim str As String
str = StrReverse$("Basic")
```

**Keterangan:**

Pada contoh di atas, *string* "Basic" akan dibalik perkarakter, sehingga akan dihasilkan *string* "cisaB".

## 17. Trim\$()

Digunakan untuk menghapus karakter spasi (kosong) pada kedua ujung kiri dan kanan *string*.

**Syntax dasar:**

```
Trim$(string)
```

Contoh:

```
Dim str As String
str = " Visual Basic "
MsgBox Trim$(str)
```

**Keterangan:**

Pada contoh di atas, karakter spasi kosong pada kedua ujung *string* " Visual Basic " akan dihapus, sehingga akan dihasilkan *string* "Visual Basic".

## 18. LTrim\$()

Digunakan untuk menghapus karakter spasi (kosong) pada ujung kiri *string*.

**Syntax dasar:**

```
LTrim$(string)
```

Contoh:

```
Dim str As String
str = " Visual Basic "
MsgBox LTrim$(str)
```

**Keterangan:**

Pada contoh di atas, karakter spasi kosong pada ujung kiri *string* " Visual Basic " akan dihapus, sehingga menghasilkan *string* "Visual Basic ".

## 19. RTrim\$()

Digunakan untuk menghapus karakter spasi (kosong) pada ujung kanan *string*.

**Syntax dasar:**

```
RTrim$(string)
```

Contoh:

```
Dim str As String
str = " Visual Basic "
MsgBox RTrim$(str)
```

**Keterangan:**

Pada contoh di atas, karakter spasi kosong pada ujung kanan *string* " Visual Basic " akan dihapus, sehingga menghasilkan *string* " Visual Basic".

## 20. Asc()

Digunakan untuk menghasilkan nilai *integer* dari karakter ASCII pertama pada suatu *string*.

**Syntax dasar:**

```
Asc(string)
```

Contoh:

```
Dim str As String
str = "Visual Basic"
MsgBox Asc(str)
```

**Keterangan:**

Karakter ASCII pertama dari *string* "Visual Basic" adalah "V". Pada contoh di atas akan dihasilkan nilai *integer* dari karakter "V", yaitu 86.

## 21. Chr\$()

Digunakan untuk menghasilkan karakter ASCII dari bilangan *integer* 0-255.

**Syntax dasar:**

```
Chr$(charcode)
```

Contoh:

```
Dim str As String
str = Chr$(86)
MsgBox (str)
```

**Keterangan:**

Pada contoh di atas akan dihasilkan karakter dari bilangan integer 86, yaitu karakter "V".

## 22. DateValue()

Digunakan untuk menghasilkan informasi waktu/tanggal tertentu.

**Syntax dasar:**

```
DateValue(date)
```

Contoh 1:

```
Dim tgl As Date
tgl = DateValue(Now)
MsgBox tgl
```

Contoh 2:

```
Dim tgl As Date
tgl = DateValue("Juni 20, 2010")
MsgBox tgl
```

## 23. TimeValue()

Digunakan untuk menghasilkan informasi jam tertentu.

**Syntax dasar:**

```
TimeValue(time)
```

Contoh 1:

```
Dim jam As Date
jam = TimeValue(Now)
MsgBox jam
```

Contoh 2:

```
Dim jam As Date
jam = TimeValue("16:35:17 PM")
MsgBox jam
```

## 24. WeekDay()

Digunakan untuk menghasilkan informasi urutan hari dalam seminggu.

**Keterangan:**

Parameter dapat diisi dengan:

```
vbSunday, -> 1
vbMonday, -> 2
vbTuesday, -> 3
vbWednesday, -> 4
vbThursday, -> 5
vbFriday, -> 6
vbSaturday, -> 7
```

Contoh:

```
MsgBox Weekday(Now)
MsgBox Weekday(vbWednesday)
```

## 25. WeekDayName()

Digunakan untuk menghasilkan informasi nama hari dalam seminggu.

Contoh:

```
MsgBox WeekdayName(Weekday(Now))
MsgBox WeekdayName(6)
```

## 26. Month()

Digunakan untuk menghasilkan informasi urutan bulan dalam setahun.

Contoh:

```
MsgBox Month(Now)
```

## 27. MonthName()

Digunakan untuk menghasilkan informasi nama bulan dalam setahun.

Contoh:

```
MsgBox MonthName(Month(Now))
```

**28. Day()**

Digunakan untuk menghasilkan informasi urutan hari dalam sebulan.

Contoh:

```
MsgBox Day (Now)
```

**29. Year()**

Digunakan untuk menghasilkan informasi tahun.

Contoh:

```
MsgBox Year (Now)
```

**30. Hour()**

Digunakan untuk menghasilkan informasi jam dalam sehari.

Contoh:

```
MsgBox Hour (Now)
```

**31. Minute()**

Digunakan untuk menghasilkan informasi menit dalam sejam.

Contoh:

```
MsgBox Minute (Now)
```

**32. Second()**

Digunakan untuk menghasilkan informasi detik dalam semenit.

Contoh:

```
MsgBox Second (Now)
```

**33. TimeSerial()**

Digunakan untuk menggabungkan informasi waktu jam, menit, dan detik.

Contoh:

```
Dim Jam As Integer
Dim Menit As Integer
Dim Detik As Integer
Dim Waktu_Jam As Date
```

```
Jam = Hour(Now)
Menit = Minute(Now)
Detik = Second(Now)
Waktu_Jam = TimeSerial(Jam, Menit, Detik)
MsgBox Waktu_Jam
```

**34. InputBox()**

Digunakan sebagai kotak dialog untuk memasukkan nilai tertentu yang dibutuhkan program.

**Syntax dasar:**

```
variable = InputBox (prompt[,title]
                    [,default])
```

Contoh:

```
Dim bil As Integer
bil = InputBox("Masukkan bilangan bulat: ", _
              "Input", 1)
If bil Mod 2 = 1 Then
    MsgBox "Bilangan ganjil"
Else
    MsgBox "Bilangan genap"
End If
```

**35. Dir\$()**

Digunakan menyatakan sebuah direktori. Statemen ini juga dapat digunakan untuk pencarian file.

**Syntax dasar:**

```
stringvar = Dir$(path[,attributes])
```

Contoh:

```
If Dir$("C:/temp.log") = "" Then
    MsgBox "File tidak ditemukan"
Else
    MsgBox "File ditemukan"
End If
```

### 36. FileCopy

Digunakan untuk menyalin (*copy*) sebuah *file* menjadi *file* lain.

**Syntax dasar:**

FileCopy source, destination

Contoh:

**FileCopy** "C:/temp.log", "C:/temp2.log"

### 37. Kill

Digunakan untuk menghapus (*delete*) sebuah *file*.

**Syntax dasar:**

Kill pathname

Contoh:

**Kill** "C:/temp2.log"

### 38. Name

Digunakan untuk mengubah nama (*rename*) sebuah *file*.

**Syntax dasar:**

Name oldpathname As newpathname

Contoh:

**Name** "C:/temp2.log" As "C:/temp\_old.log"

### 39. Mkdir

Digunakan untuk membuat *folder* baru.

**Syntax dasar:**

Mkdir path

Contoh:

**Mkdir** "c:\DataVB\"

### 40. Rmdir

Digunakan untuk menghapus *folder*.

**Syntax dasar:**

Rmdir path

Contoh:

**Rmdir** "c:\DataVB\"